X3D Graphics for Web Authors

Chapter 9

Event Utilities and Scripting

Action is eloquence.

William Shakespeare, *Coriolanus*, Act III Scene II

Contents

Chapter Overview

Overview: Event Utilities and Scripting

Event utility nodes simplify data-type conversion, creation of some ROUTE connections is easier

Sequencer nodes similar to Interpolator functions

- Interpolators: continuous (floating point) outputs
- Sequencers: discrete (boolean, integer) outputs

Numerous event-utility nodes for thoroughness

- BooleanFilter, BooleanSequencer, BooleanToggle, BooleanTrigger
- IntegerSequencer, IntegerTrigger, TimeTrigger

Script node encapsulates ECMAScript, Java code

Concepts

Event utilities motivation 1

Event ROUTE connections are fundamental to X3D animation

Every node in the scene graph can be an animation candidate

- Many nodes have output fields to produce values
- Most nodes have input fields that can be changed by run-time events
- Can connect each output-to-input pair via a ROUTE

Certain common animation challenges led to development of event-utility node extensions to original X3D/VRML scene graph

Event utilities motivation 2

Modification of field values worked well in X3D v3.0, but some animation chains were difficult to create unless Script nodes were used

- Boolean logic combinations
- Converting time events to boolean, and vice versa
- Producing an animation sequence of integer values

Type conversion and logical operations were among most common needs for Script nodes

Adding event utility nodes provides much better coverage for needed event-chain connections

Functional summary, comparison

- BooleanFilter and BooleanToggle simplify the negation, combination of boolean true/false logic
- BooleanSequencer and IntegerSequencer are similar to interpolator nodes, but produce single discrete values one at a time, rather than continuous stream of changing floating-point values
- BooleanTrigger, IntegerTrigger, TimeTrigger each produce a single typed-value response after receiving a triggering input
- Script node can encapsulate arbitrary functionality, using either ECMAScript (JavaScript) or Java, with resulting capabilities similar to HTML scripting

Declarative programming

Language is "declarative" if it describes solutions based on problem aspects, rather than directing step-by-step algorithm how to solve it

- Example: HTML for web pages
- HTML file declares *what* should appear on a page (formatted text, images, links, etc.) but does not specify *how* a browser accomplishes that rendering

Some programming languages are declarative

- Prolog (Programming in Logic) for logical inference
- Expert systems containing rules, relationships
- Extensible Stylesheet Language for Transformations (XSLT) processes XML inputs

Imperative programming

Imperative programming languages define an ordered sequence of steps to be followed

- Basic, C, C++, Fortran, Java, many other languages

Imperative tasks typically follow an algorithmic program or list of procedures that

- Accept input values
- Compute state-variable results and save values
- Produce an appropriate output

Algorithms start from a given state, iterate through task lists, and eventually terminate

Declarative compared to imperative

Declarative programs usually constructed as a top-down decomposition approach

- For each part of problem, solution is defined
- Control implicitly driven by input, perhaps recursive

Imperative programs usually constructed as a bottom-up directive approach

- Much more like "do this, then do that, repeat, etc."

Declarative approach is less common than imperative approach, but is quite powerful

- Often preferred for many types of problems

Example: dominoes ready to fall

X3D scene graph is declarative

X3D design is declarative, because

- presence of various sensor and interpolator nodes,
- plus the ROUTE connections within scene graph,
- defines how scene behaves, reacting to event inputs

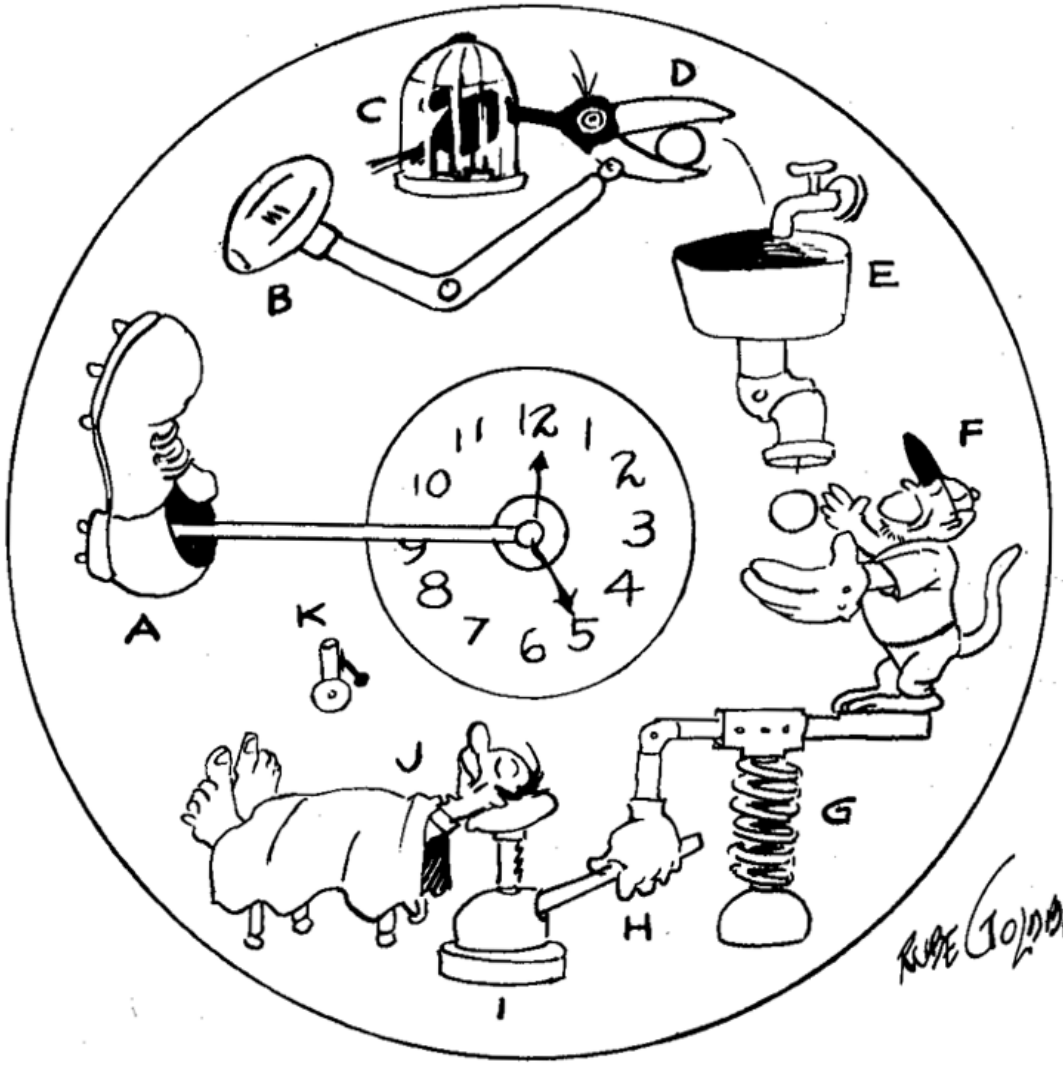
Can be thought of as a cause-and-effect chain

- i.e. row of dominoes ready to tip, or
- perhaps some kind of Rube Goldberg machine

Declarative thus indicates that action/reaction relationships are predeclared

- Operates after loading and clock, interaction begins
- Input stimulus needed before output is produced

Example: Rube Goldberg machine



RUBE GOLDBERG Alarm Clock

Shoe (A)
Kicks football (B)
Opening mouth of bird (C)
And dropping baseball (D)
Into sink (E)
Baseball goes through drain
and falls into glove (F)
Causing spring (G)
To move hand (H)
Which works jack (I)
Lifting sleeper (J)
Into vertical position -
Bell (K) Rings to make sure
sleeper gets up!

Script node motivation: imperative

Script node encapsulates *imperative* programs, wrapping them in an event-driven interface suitable for connection to the *declarative* approach of the X3D scene graph

- Thus an impedance match between imperative and declarative programming paradigms

Embedding (often small) chunks of imperative script code within a declarative event-passing X3D animation chain is powerful

- Algorithms of arbitrary complexity can be built
- Well-defined rules align these 2 different paradigms
- Similar motivations for HTML use of JavaScript

Common functionality

Event utility nodes and Script nodes can appear in the scene graph anywhere that other children nodes can appear

- Reminder: not within Shape, for example

Must have a DEF name in order to ROUTE input and output values

- Not much point to these nodes if not connected...

Like interpolator nodes, their position in scene graph has no effect on their functionality

- Must precede any ROUTE that references them

Sequencer nodes

BooleanSequencer and IntegerSequencer are both of X3DSequencerNode type

Similar to interpolator nodes, which define functions producing continuous stream of steadily changing floating-point values

Sequencer outputs produce single discrete values one at a time, as needed

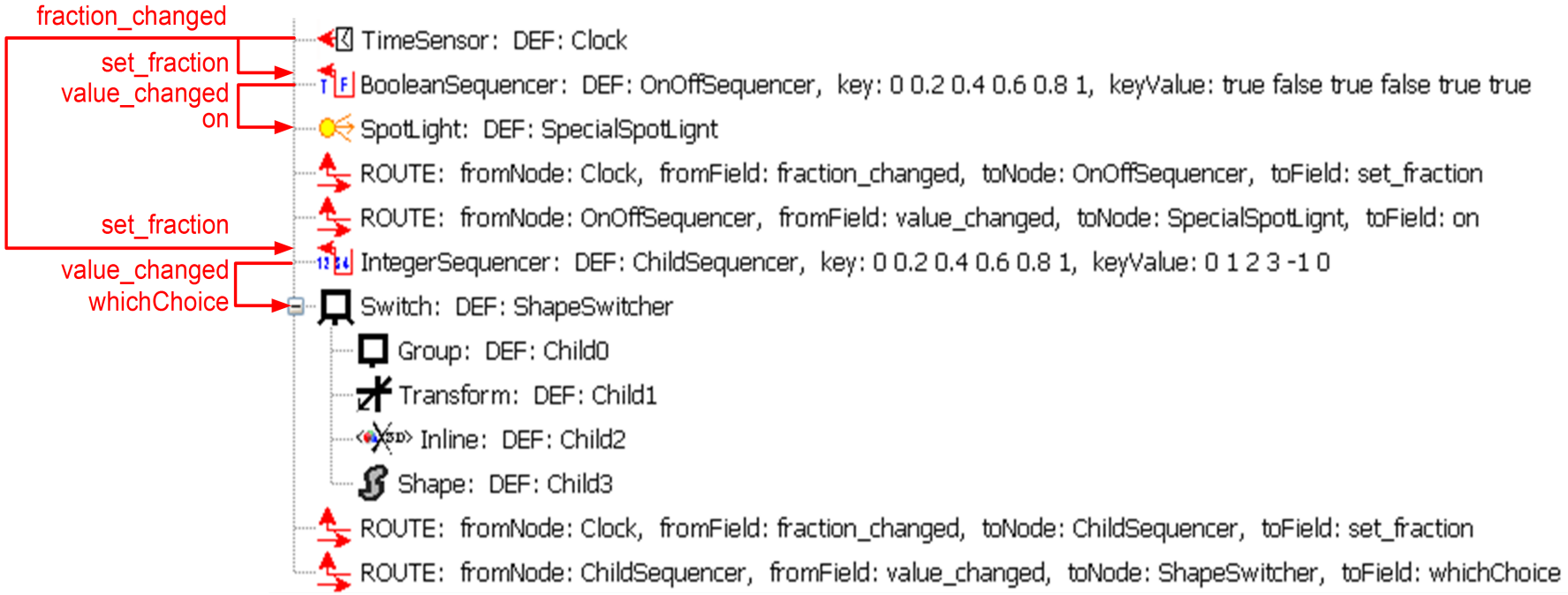
Example: count -1..10 to change a Switch node

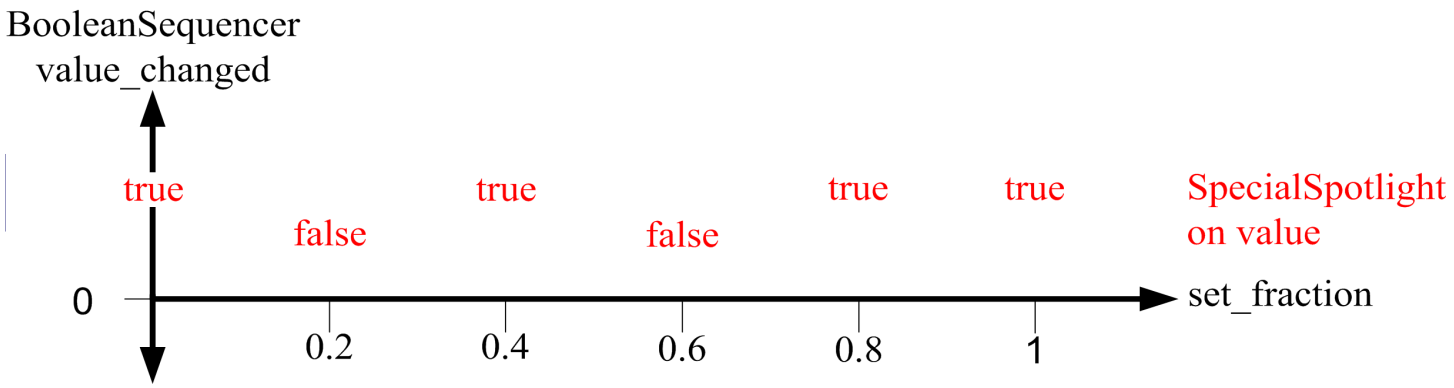
- No need to repeatedly send '1' values, or '2' values, etc. in between times when actual change occurs

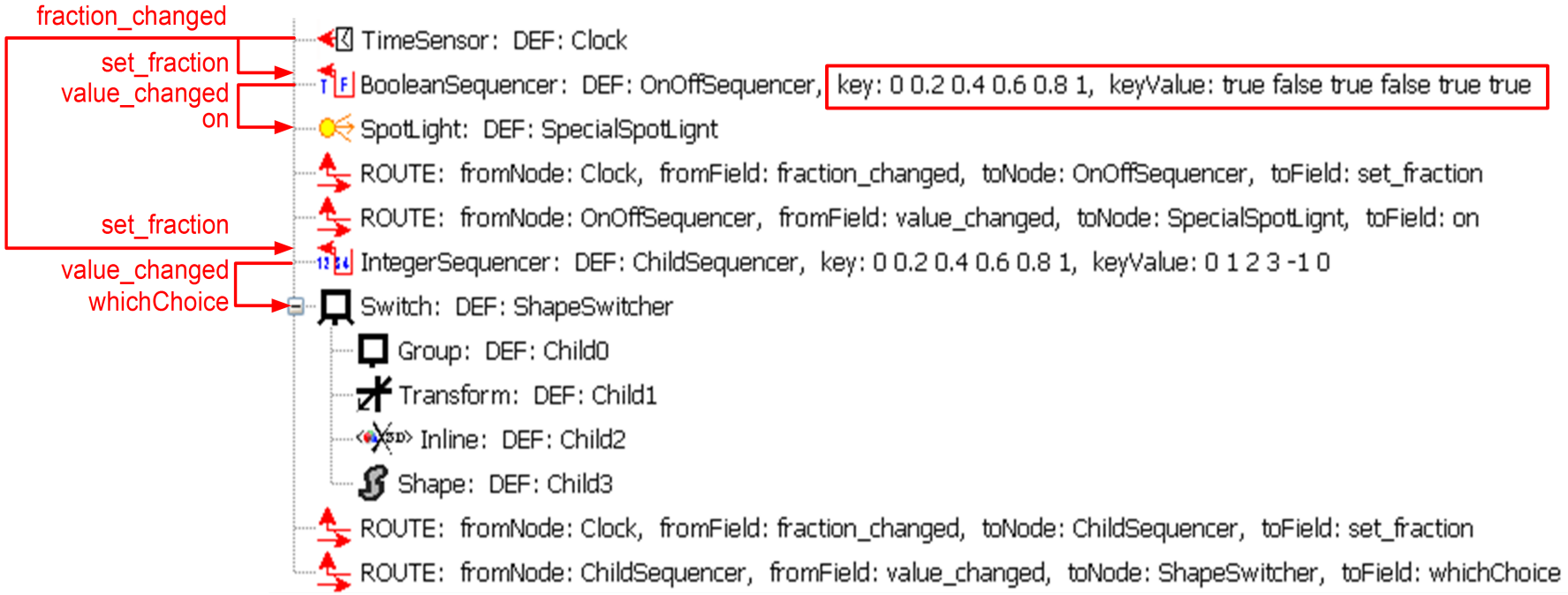
Example: boolean, integer sequencers

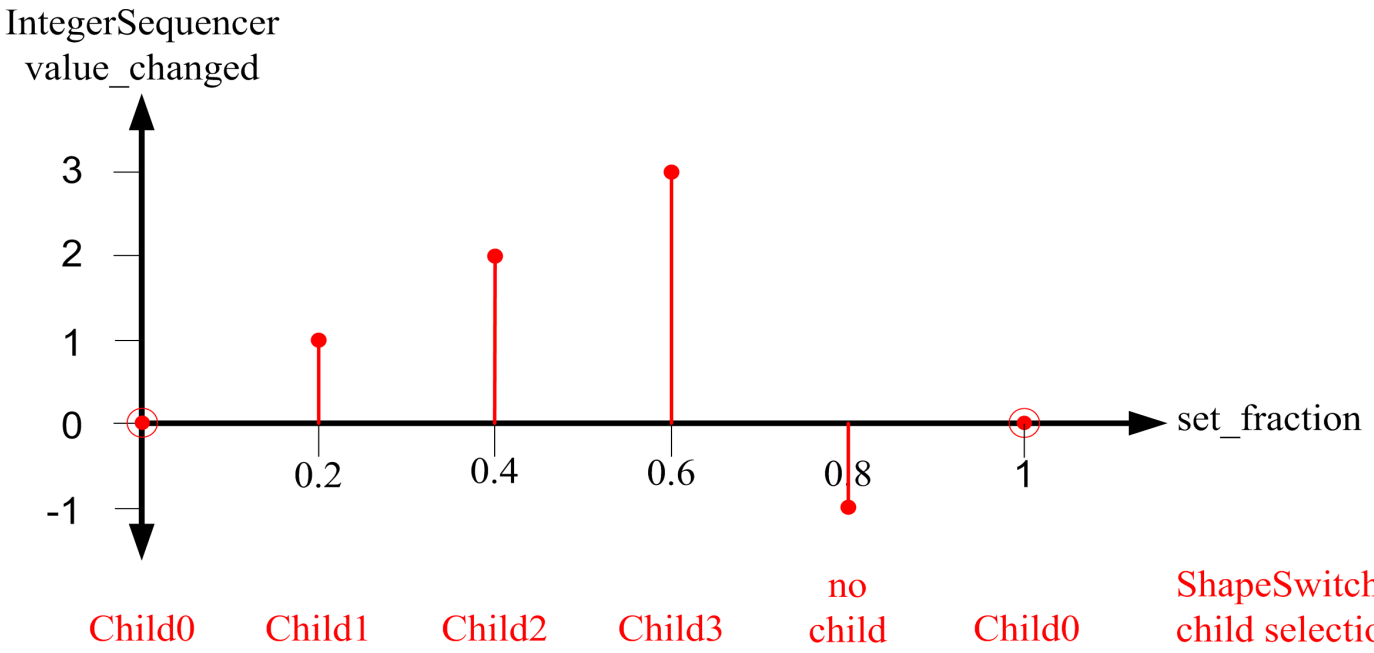
Following example illustrates coordinated use of BooleanSequencer, IntegerSequencer nodes

- Note common TimeSensor node (named Clock)



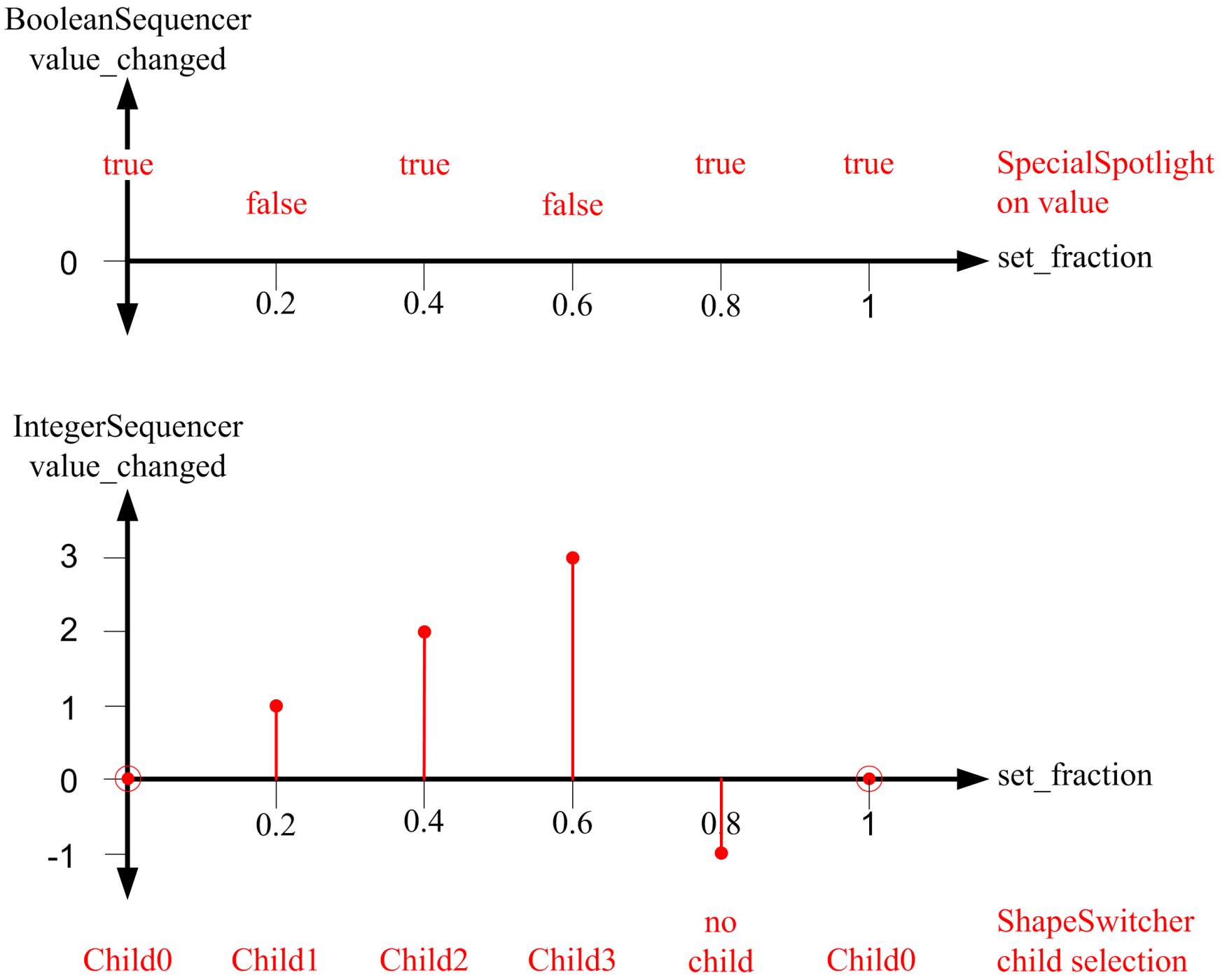

BooleanSequencer
value_changed
true
false
true
false
true
true
SpecialSpotlight
on value
0
set_fraction
0.2
0.4
0.6
0.8
1


fraction_changed
set_fraction
value_changed
on
set_fraction
value_changed
whichChoice
TimeSensor: DEF: Clock
BooleanSequencer: DEF: OnOffSequencer, key: 0 0.2 0.4 0.6 0.8 1, keyValue: true false true false true true
SpotLight: DEF: SpecialSpotLignt
ROUTE: fromNode: Clock, fromField: fraction_changed, toNode: OnOffSequencer, toField: set_fraction
ROUTE: fromNode: OnOffSequencer, fromField: value_changed, toNode: SpecialSpotLignt, toField: on
IntegerSequencer: DEF: ChildSequencer, key: 0 0.2 0.4 0.6 0.8 1, keyValue: 0 1 2 3 -1 0
Switch: DEF: ShapeSwitcher
Group: DEF: Child0
Transform: DEF: Child1
Inline: DEF: Child2
Shape: DEF: Child3
ROUTE: fromNode: Clock, fromField: fraction_changed, toNode: ChildSequencer, toField: set_fraction
ROUTE: fromNode: ChildSequencer, fromField: value_changed, toNode: ShapeSwitcher, toField: whichChoice



fraction_changed
set_fraction
value_changed
on
set_fraction
value_changed
whichChoice

TimeSensor: DEF: Clock
BooleanSequencer: DEF: OnOffSequencer, key: 0 0.2 0.4 0.6 0.8 1, keyValue: true false true false true true
SpotLight: DEF: SpecialSpotLignt
ROUTE: fromNode: Clock, fromField: fraction_changed, toNode: OnOffSequencer, toField: set_fraction
ROUTE: fromNode: OnOffSequencer, fromField: value_changed, toNode: SpecialSpotLignt, toField: on
IntegerSequencer: DEF: ChildSequencer, key: 0 0.2 0.4 0.6 0.8 1, keyValue: 0 1 2 3 -1 0
Switch: DEF: ShapeSwitcher
Group: DEF: Child0
Transform: DEF: Child1
Inline: DEF: Child2
Shape: DEF: Child3
ROUTE: fromNode: Clock, fromField: fraction_changed, toNode: ChildSequencer, toField: set_fraction
ROUTE: fromNode: ChildSequencer, fromField: value_changed, toNode: ShapeSwitcher, toField: whichChoice


BooleanSequencer
value_changed
true
false
true
false
true
true
SpecialSpotlight
on value
0
0.2
0.4
0.6
0.8
1
set_fraction
IntegerSequencer
value_changed
3
2
1
0
-1
0.2
0.4
0.6
0.8
1
set_fraction
Child0
Child1
Child2
Child3
no
child
Child0
ShapeSwitcher
child selection

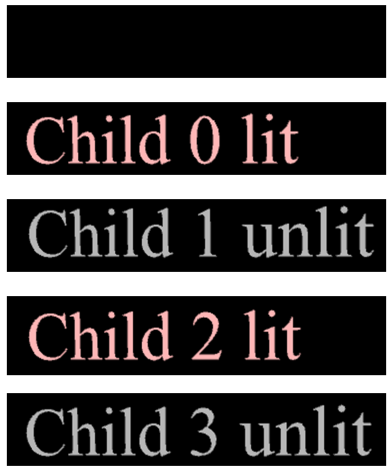
BooleanSequencerIntegerSequence.x3d - Editor

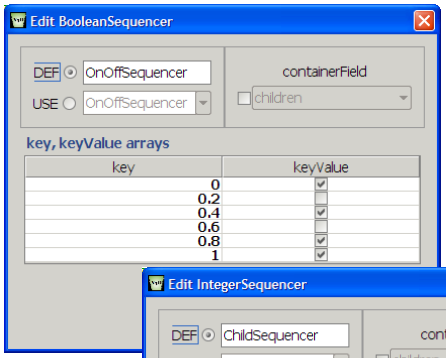
BooleanSequencerIntegerSequence.x3d

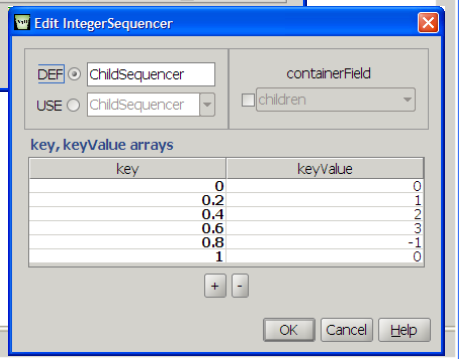
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile="Immersive" version="3.1"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema-instance" xsd:noNamespaceSchemaLocation="http://www.web3d.org/specifications/x3d-3.1.xsd">
    <head>
        <meta content="BooleanSequencerIntegerSequencer.x3d" name="title"/>
        <meta content="Show synchronized use of BooleanSequencer and IntegerSequencer nodes" name="description"/>
        <meta content="Don Brutzman" name="creator"/>
        <meta content="6 September 2006" name="created"/>
        <meta content='24 February 2008' name='modified'/>
        <meta content="http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d"
              name="identifier"/>
        <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
        <meta content="../../license.html" name="license"/>
    </head>
    <Scene>
        <TimeSensor DEF="Clock" cycleInterval="5" loop="true"/>
        <BooleanSequencer DEF="OnOffSequencer" key="0 0.2 0.4 0.6 0.8 1" keyValue="true false true false true true"/>
        <SpotLight DEF="SpecialSpotLight" color="1 0.0 0.0" location='0 0 10'/>
        <ROUTE fromField="fraction_changed" fromNode="Clock" toField="set_fraction" toNode="OnOffSequencer"/>
        <ROUTE fromField="value_changed" fromNode="OnOffSequencer" toField="on" toNode="SpecialSpotLight"/>
        <IntegerSequencer DEF="ChildSequencer" key="0 0.2 0.4 0.6 0.8 1" keyValue="0 1 2 3 -1 0"/>
        <Transform translation="-1.75 0.5 0">
            <Switch DEF="ShapeSwitcher">
                <ROUTE fromField="fraction_changed" fromNode="Clock" toField="set_fraction" toNode="ChildSequencer"/>
                <ROUTE fromField="value_changed" fromNode="ChildSequencer"  toField="whichChoice" toNode="ShapeSwitcher"/>
                <Shape DEF="Child0">
                    <Text string='Child 0 lit'/>
                    <Appearance DEF="GreyAppearance">
                        <Material diffuseColor="0.7 0.7 0.7"/>
                    </Appearance>
                </Shape>
                <Shape DEF="Child1">
                    <Text string='Child 1 unlit'/>
                    <Appearance USE="GreyAppearance"/>
                </Shape>
                <Shape DEF="Child2">
                    <Text string='Child 2 lit'/>
                    <Appearance USE="GreyAppearance"/>
                </Shape>
                <Shape DEF="Child3">
                    <Text string='Child 3 unlit'/>
                    <Appearance USE="GreyAppearance"/>
                </Shape>
            </Switch>
        </Transform>
    </Scene>
</X3D>
```

BooleanSequencer turns SpotLight on, off

IntegerSequencer switches choice of Text







Single TimeSensor clock drives two synchronized BooleanSequencer, IntegerSequencer output streams

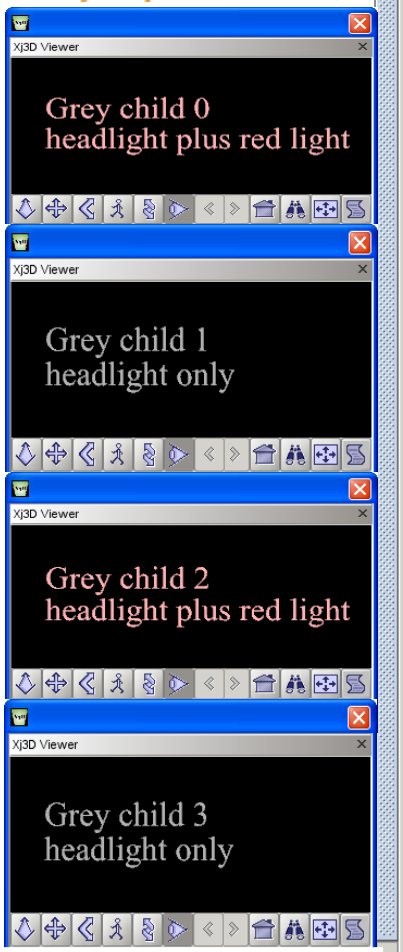
18:118 INS

BooleanSequencerIntegerSequencer.x3d - Editor

BooleanSequencerIntegerSequencer.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/sp
  <head>
    <meta content='BooleanSequencerIntegerSequencer.x3d' name='title'/>
    <meta content='Show synchronized use of BooleanSequencer and IntegerSequencer nodes for SpotLight enabled and Switch whichChoice control, respe
    <meta content='Don Brutzman' name='creator'/>
    <meta content='6 September 2006' name='created'/>
    <meta content='30 March 2009' name='modified'/>
    <meta content='http://x3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d' name='
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../../license.html' name='license'/>
  </head>
  <Scene>
    <NavigationInfo headlight='true'/>
    <TimeSensor DEF='Clock' cycleInterval='5' loop='true'/>
    <BooleanSequencer DEF='OnOffSequencer' key='0 0.2 0.4 0.6 0.8 1' keyValue='true false true false true true'/>
    <SpotLight DEF='SpecialSpotLight' color='1 0.0 0.0' location='0 0 10'/>
    <ROUTE fromField='fraction_changed' fromNode='Clock' toField='set_fraction' toNode='OnOffSequencer'/>
    <ROUTE fromField='value_changed' fromNode='OnOffSequencer' toField='on' toNode='SpecialSpotLight'/>
    <IntegerSequencer DEF='ChildSequencer' key='0 0.2 0.4 0.6 0.8 1' keyValue='0 1 2 3 -1 0'/>
    <Transform translation='-3.5 1 0'>
      <Switch DEF='ShapeSwitcher'>
        <ROUTE fromField='fraction_changed' fromNode='Clock' toField='set_fraction' toNode='ChildSequencer'/>
        <ROUTE fromField='value_changed' fromNode='ChildSequencer' toField='whichChoice' toNode='ShapeSwitcher'/>
        <Shape DEF='Child0'>
          <Text string='"Grey child 0" "headlight plus red light"'/>
          <Appearance DEF='GreyAppearance'>
            <Material diffuseColor='0.7 0.7 0.7'/>
          </Appearance>
        </Shape>
        <Shape DEF='Child1'>
          <Text string='"Grey child 1" "headlight only"'/>
          <Appearance USE='GreyAppearance'/>
        </Shape>
        <Shape DEF='Child2'>
          <Text string='"Grey child 2" "headlight plus red light"'/>
          <Appearance USE='GreyAppearance'/>
        </Shape>
        <Shape DEF='Child3'>
          <Text string='"Grey child 3" "headlight only"'/>
          <Appearance USE='GreyAppearance'/>
        </Shape>
      </Switch>
    </Transform>
  </Scene>
```

whichChoice



Single TimeSensor clock drives two synchronized BooleanSequencer, IntegerSequencer output streams

Scene text revised March 2009

1:1 INS

Fan in, fan out

Multiple-event ***fan out***

- More than one ROUTE may connect a single output field to other nodes

Multiple-event ***fan in***

- More than one ROUTE may also expose a single input field to the output of more than one event-producing node
- Can be error prone and non-deterministic if events are allowed to occur simultaneously

X3D Nodes and Examples

BooleanFilter node

BooleanFilter allows selective filtering of true/false event pairs, typically sent by sensors

Example: choose among TouchSensor events

- sends *isActive, isOver* true when selected, then
- sends *isActive, isOver* false when deselected

Field definitions

- *set_boolean* is input value to be filtered
- *inputTrue*, *inputFalse* each output true if condition met
- *inputNegate* sends value opposite to *set_boolean*
- No initializable fields to edit

BooleanFilterPumpHouse.x3d - Editor

BooleanFilterPumpHouse.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://ww
    <head>
        <meta content='BooleanFilterPumpHouse.x3d' name='title'/>
        <meta content='A BooleanFilter node controls the animation of orbiting cones near the pump house.' name='description'/>
        <meta content='Todd Gagnon and Mark A. Boyd' name='authors'/>
        <meta content='Xeena VRML importer' name='translator'/>
        <meta content='8 June 1998' name='created'/>
        <meta content='20 December 2002' name='imported'/>
        <meta content='24 February 2008' name='modified'/>
        <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference'/>
        <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanFilterPumpHouse.x3d'
        <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
        <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator'/>
        <meta content='../license.html' name='license'/>
    </head>
    <Scene>
        <Background skyColor='1 1 1'/>
        <NavigationInfo type='"FLY" "ANY"'/>
        <Viewpoint description='Click pump house to activate' position='2 1 9'/>
        <TimeSensor DEF='Timer' cycleInterval='2' enabled='false' loop='true' startTime='1'/>
        <OrientationInterpolator DEF='Rotator' key='0 .33 .67 1' keyValue='0 0 1 0 0 0 1 2 0 0 1 4 0 0 1 6.28'/>
        <BooleanFilter DEF='Filter'/>
        <Transform DEF='SpinningColoredCones' translation='3 2 1'>
            <Transform>
            <Transform>
            <Transform>
        </Transform>
        <ROUTE fromField='inputTrue' fromNode='Filter' toField='enabled' toNode='Timer'/>
        <ROUTE fromField='fraction_changed' fromNode='Timer' toField='set_fraction' toNode='Rotator'/>
        <ROUTE fromField='value_changed' fromNode='Rotator' toField='rotation' toNode='SpinningColoredCones'/>
        <Group>
            <TouchSensor DEF='RunPump' description='touch to activate' enabled='true'/>
            <ROUTE fromField='isActive' fromNode='RunPump' toField='set_boolean' toNode='Filter'/>
            <!-- pump house geometry follows -->
            <Transform scale='0.91 0.6 0.3' translation='0.8 -0.65 0.5'>
                <Shape>
                    <Appearance>
                        <Material diffuseColor='0.749 0.694 0.651'/>
                    </Appearance>
                    <Cylinder bottom='false' top='false'/>
```

Filtering out the *isActive* false events enables this TimeSensor clock to continue looping, once started

enabled

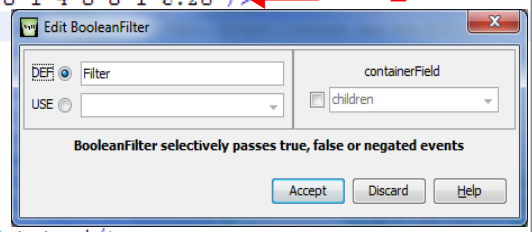
inputTrue

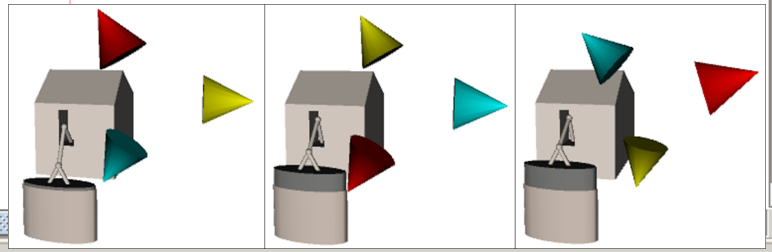
set_boolean

isActive

fraction_changed
set_fraction

value_changed
set_rotation





24:23 INS

BooleanFilter	BooleanFilter selectively passes true, false or negated events.
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
set_boolean	**[set_boolean: accessType inputOnly, type SFBool (true|false) #FIXED ""]** set_boolean is the input value to be filtered.
inputTrue	**[inputTrue: accessType outputOnly, type SFBool (true|false) #FIXED ""]** inputTrue only passes a true value, when set_boolean input is true.
inputFalse	**[inputFalse: accessType outputOnly, type SFBool (true|false) #FIXED ""]** inputFalse only passes a false value, when set_boolean is false.
inputNegate	**[inputNegate: accessType outputOnly, type SFBool (true|false) #FIXED ""]** inputNegate provides opposite value by negating set_boolean input.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

BooleanSequencer node

BooleanSequencer contains a *keyValue* array of boolean values that can be triggered by input *set_fraction* values for corresponding *key* array

As with interpolator nodes, output field is named *value_changed*

Useful fields: node triggering via boolean events

- SFBool inputOnly *next:* send next *keyValue*
- SFBool inputOnly *previous:* send prior *keyValue*
- Provides a helpful alternative to using the more typical TimeSensor *fraction_changed*

BooleanSequencerPumpHouse.x3d - Editor

BooleanSequencerPumpHouse.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.
  <head>
    <meta content='BooleanSequencerPumpHouse.x3d' name='title'/>
    <meta content='A BooleanSequencer node intermittently interrupts animation of the pump house.' name='description'/>
    <meta content='Leonard Daly, Don Brutzman, Todd Gagnon and Mark A. Boyd' name='authors'/>
    <meta content='Xeena VRML importer' name='translator'/>
    <meta content='8 June 1998' name='created'/>
    <meta content='20 December 2002' name='imported'/>
    <meta content='8 October 2007' name='modified'/>
    <meta content='KelpTank.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerPumpHouse.x3d' name='ic
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='touch and hold to interrupt pump operation' orientation='-0.969 0.239
    <Group DEF='PumpHouseGroup'>
      <Transform>
      <Group>
        <Shape>
        <TouchSensor DEF='InterruptPump' description='touch and hold to interrupt pump operation'
      </Group>
    </Group>
    <Group>
      <Transform scale='0.5 0.5 0.5' translation='1.0 1.1 -1.5'>
        <Transform DEF='PISTON'>
          <PositionInterpolator DEF='PISTONPath' key='0.0 0.3 0.32 0.5 0.75 1.0' keyValue='-0.4 -2.3 4.0 -0.4 -1.5 4.0 -0.4 -1.5 4.0 -0.4 -2.
          <TimeSensor DEF='PISTONClock' cycleInterval='5.5' enabled='true' loop='true'/>
          <BooleanSequencer DEF="OnOffSequencer" key="0 0.25 0.33 0.5 0.625 0.67 0.75 0.9 1" keyValue="true false true false true false true
          <TimeSensor DEF='InterruptTimer' cycleInterval='6' enabled='true' loop='true' startTime='1'/>
          <ROUTE fromField='value_changed' fromNode='PISTONPath' toField='set_translation' toNode='PISTON'/>
          <ROUTE fromField='fraction_changed' fromNode='PISTONClock' toField='set_fraction' toNode='PISTONPath'/>
          <ROUTE fromField='value_changed' fromNode='OnOffSequencer' toField='enabled' toNode='PISTONClock'/>
          <ROUTE fromField='fraction_changed' fromNode='InterruptTimer' toField='set_fraction' toNode='OnOffSequencer'/>
          <ROUTE fromField='isActive' fromNode='InterruptPump' toField='enabled' toNode='InterruptTimer'/>
          <Transform scale='1.8 1.2 0.6' translation='0.0 -0.2 0.0'>
            <Shape>
              <Appearance>
                <Material diffuseColor='0.427 0.427 0.42'/>
```

TouchSensor.
isActive

Edit BooleanSequencer

DEF OnOffSequencer
USE
containerField
children
key, keyValue arrays

#	key	keyValue
0	0	☑
1	0.25	☐
2	0.33	☑
3	0.5	☐
4	0.625	☑
5	0.67	☐
6	0.75	☑
7	0.9	☐
8	1	☑

Copy Add Remove Append: commas, line breaks
set uniform key spacing
Accept Discard Help

TimeSensor drives BooleanSequencer outputs to provide steady stream of true/false values

48:141 INS

BooleanSequencer	**BooleanSequencer generates periodic discrete Boolean values that can be ROUTEd to other Boolean attributes. Typical input: ROUTE someTimeSensor.fraction_changed TO someInterpolator.set_fraction Typical output: ROUTE someInterpolator.value_changed TO destinationNode.set_attribute.**
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
key	**[key: accessType inputOutput, type MFFloat CDATA #IMPLIED]** Definition parameters for linear-interpolation function time intervals, in increasing order and corresponding to keyValues. **Hint:** number of keys must match number of keyValues!
keyValue	**[keyValue: accessType inputOutput, type MFBool (true|false) CDATA, SFString for VRML 97 #IMPLIED]** Output values for linear interopolation, each corresponding to time-fraction keys. **Hint:** number of keys must match number of keyValues!
set_fraction	**[set_fraction: inputOnly type SFFloat CDATA #FIXED ""]** set_fraction selects input key for corresponding keyValue output.
value_changed	**[value_changed: accessType outputOnly, type SFBool (true|false) #FIXED ""]** Single intermittent output value determined by current key time and corresponding keyValue pair.
previous	**[previous: accessType inputOnly, type SFBool (true|false) "0"]** Trigger previous output value in keyValue array. **Hint:** loops from first to last if necessary.
next	**[next: accessType inputOnly, type SFBool (true|false) "0"]** Trigger next output value in keyValue array. **Hint:** loops from last to first if necessary.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

BooleanToggle node

BooleanToggle remembers, can negate booleans

- Maintains inputOutput state variable named *toggle*
- Negates *toggle* value if input *set_boolean* is true
- No change occurs if input event *set_boolean* is false

Honoring input *set_boolean* true (and ignoring *set_boolean* false) is especially helpful when connected to *isActive/isOver* sensor outputs

- Example: only an *isActive* true event causes change
- Toggling occurs only on selection, rather than both selection+deselection, so user can remove pointer

Some typical event-animation chains

Typically task: author wants to toggle a value, then enable or disable an animation chain, based on user selecting an object in the scene

- TouchSensor.*isActive* → BooleanToggle.*set_boolean*
- BooleanToggle.*toggle* → TimeSensor.*enabled*

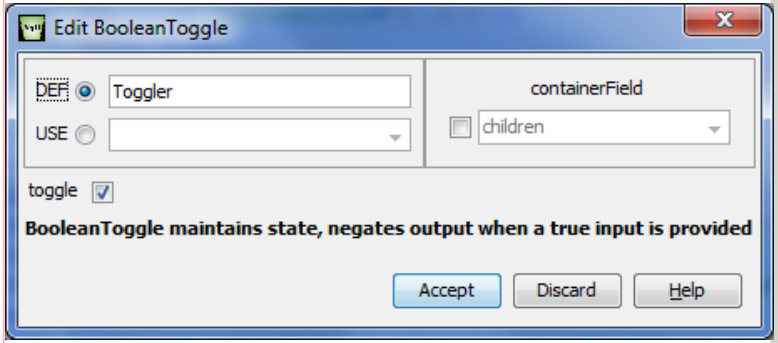
BooleanToggle can instead swap on deselection by adding an intermediate BooleanFilter node:

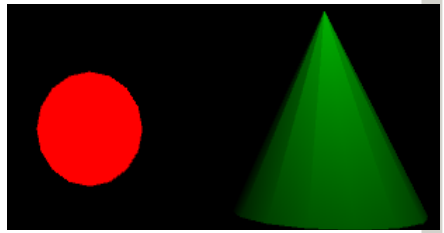
- TouchSensor.*isActive* → BooleanFilter.*set_boolean*
- BooleanFilter.*inputFalse* → BooleanToggle.*set_boolean*
- BooleanToggle.*toggle* → TimeSensor.*enabled*

BooleanToggle.x3d - Editor

BooleanToggle.x3d

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3  <X3D profile='Interactive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web
4    <head>
5      <meta content='BooleanToggle.x3d' name='title'/>
6      <meta content='A BooleanToggle button determines whether an animated Cone is jittery or not.' name='description'/>
7      <meta content='Leonard Daly and Don Brutzman' name='creator'/>
8      <meta content='9 October 2006' name='created'/>
9      <meta content='24 February 2008' name='modified'/>
10     <meta content='http://X3dGraphics.com' name='reference'/>
11     <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
12     <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
13     <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
14     <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanToggle.x3d' name='identifier'/>
15     <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
16     <meta content='../license.html' name='license'/>
17   </head>
18   <Scene>
19     <Transform rotation='1 0 0 1.57'>
20       <Shape>
26       <TouchSensor DEF='TouchButton' description='touch to activate'/>
27     </Transform>
28     <Transform translation='2 0 0'>
29       <Transform DEF='ShakingCone'>
30         <Shape>
36       </Transform>
37     </Transform>
38     <BooleanToggle DEF='Toggler'/>
39     <ROUTE fromField='isActive' fromNode='TouchButton' toField='set_boolean' toNode='Toggler'/>
40     <BooleanFilter DEF='Tee'/>
41     <ROUTE fromField='toggle_changed' fromNode='Toggler' toField='set_boolean' toNode='Tee'/>
42     <TimeTrigger DEF='PauseOn'/>
43     <ROUTE fromField='inputTrue' fromNode='Tee' toField='set_boolean' toNode='PauseOn'/>
44     <TimeTrigger DEF='PauseOff'/>
45     <ROUTE fromField='inputFalse' fromNode='Tee' toField='set_boolean' toNode='PauseOff'/>
46     <TimeSensor DEF='Timer' cycleInterval='.15' enabled='true' loop='true' startTime='1'/>
47     <ROUTE fromField='triggerTime' fromNode='PauseOn' toField='pauseTime' toNode='Timer'/>
48     <ROUTE fromField='triggerTime' fromNode='PauseOff' toField='resumeTime' toNode='Timer'/>
49     <PositionInterpolator DEF='Shaker' key='0 .25 .50 .75 1' keyValue='0 0 0 .4 .2 -.3 -.2 .3 .1 -.1 -.2 .3 0 0 0'/>
50     <ROUTE fromField='fraction_changed' fromNode='Timer' toField='set_fraction' toNode='Shaker'/>
51     <ROUTE fromField='value_changed' fromNode='Shaker' toField='translation' toNode='ShakingCone'/>
52   </Scene>
53 </X3D>
```





BooleanToggle only flips state true/false once per each selection by TouchSensor “TouchButton”

38:19 INS

BooleanToggle	BooleanToggle maintains state and negates output when a true input is provided.
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
set_boolean	**[set_boolean: accessType inputOnly, type SFBool (true|false) #FIXED ""]** If set_boolean input is true, toggle state.
toggle	**[toggle: accessType inputOutput, type SFBool (true|false) #FIXED ""]** Persistent state value that gets toggled or reset.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

BooleanTrigger node

BooleanTrigger converts input SFTime events into output SFBool *triggerTrue* output events

- Convenient type conversion, result is always *true*

May be especially helpful for connecting certain intermittent timing events:

- TimeSensor *cycleTime* or TouchSensor *touchTime*

Typical targets for output:

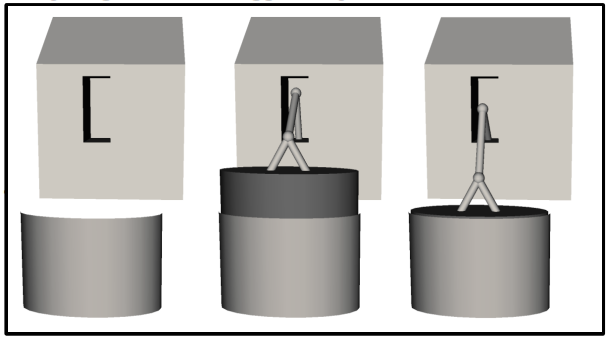
- *enabled* field for a sensor node

If a *false* event output is needed instead, ROUTE a connection via BooleanFilter *inputNegate*

BooleanTriggerPumpHouse.x3d - Editor

BooleanTriggerPumpHouse.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.
  <head>
    <meta content='BooleanTriggerPumpHouse.x3d' name='title'/>
    <meta content='A BooleanTrigger node initiates the pump house animation:  click to start.' name='description'/>
    <meta content='Positive-displacement cylinder pump to emulate breakers surge, designed and built by David Packard.' name='description'/>
    <meta content='Todd Gagnon and Mark A. Boyd' name='authors'/>
    <meta content='Xeena VRML importer' name='translator'/>
    <meta content='8 June 1998' name='created'/>
    <meta content='20 December 2002' name='imported'/>
    <meta content='25 February 2008' name='modified'/>
    <meta content='KelpTank.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanTriggerPumpHouse.x3d' name='iden
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='select pump house to activate' orientation='-0.969 0.239 0.056 0
    <Group>
      <Transform>
      <Group>
        <Shape>
          <Appearance>
          <IndexedFaceSet>
        </Shape>
        <TouchSensor DEF='RunPump' description='touch to activate' enabled='true'/>
      </Group>
    </Group>
    <Group>
      <Transform scale='0.5 0.5 0.5' translation='1.0 1.1 -1.5'>
        <Transform DEF='PistonTransform'>
          <PositionInterpolator DEF='PistonPath' key='0.0 0.3 0.32 0.5 0.75 1.0' keyValue
          <TimeSensor DEF='PistonClock' cycleInterval='5.5' enabled='false' loop='true'/>
          <BooleanTrigger DEF='TriggerStart'/>
          <ROUTE fromField='touchTime' fromNode='RunPump' toField='set_triggerTime' toNode='TriggerStart'/>
          <ROUTE fromField='triggerTrue' fromNode='TriggerStart' toField='enabled' toNode='PistonClock'/>
          <ROUTE fromField='fraction_changed' fromNode='PistonClock' toField='set_fraction' toNode='PistonPath'/>
          <ROUTE fromField='value_changed' fromNode='PistonPath' toField='set_translation' toNode='PistonTransform'/>
          <!-- remainder of pump assembly follows -->
          <Transform scale='1.8 1.2 0.6' translation='0.0 -0.2 0.0'>
```



Edit BooleanTrigger

DEF: TriggerStart

USE:

containerField: children

BooleanTrigger converts time events to boolean true events

Accept | Discard | Help

TouchSensor.
touchTime

BooleanTrigger “TriggerStart” converts touchTime to boolean, enabling TimeSensor

49:26 INS

BooleanTrigger	BooleanTrigger converts time events to boolean true events.
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
set_triggerTime	**[set_triggerTime: accessType inputOnly, type SFTime CDATA #FIXED ""]** set_triggerTime provides input time event, typical event sent is TouchSensor touchTime.
triggerTrue	**[triggerTrue: accessType outputOnly, type SFBool (true|false) #FIXED ""]** triggerTrue outputs a true value whenever a triggerTime event is received.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

IntegerSequencer node

IntegerSequencer outputs a series of SFInt32 integer values, based on a floating-point fractional input and *key, keyValue* pairs

- Continuous input similar to interpolator nodes
- Discrete output similar to BooleanSequencer

As with interpolator nodes, output field is named *value_changed*

Useful fields: node triggering via boolean events

- SFBool inputOnly *next:* send next *keyValue*
- SFBool inputOnly *previous:* send prior *keyValue*
- Provides a helpful alternative to using the more typical TimeSensor *fraction_changed*

IntegerSequencerPumpHouse.x3d - Editor

IntegerSequencerPumpHouse.x3d

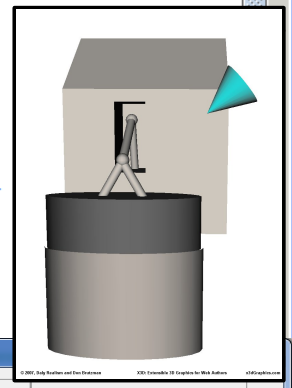
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org
  <head>
    <meta content='IntegerSequencerPumpHouse.x3d' name='title'/>
    <meta content='An IntegerSequencer node switches the display of colored cones near the pump house.' name='description'/>
    <meta content='Todd Gagnon and Mark A. Boyd' name='authors'/>
    <meta content='Xeena VRML importer, X3D-Edit 3.1, http://www.web3d.org/x3d/content/README.X3D-Edit.html' name='translator'/>
    <meta content='8 June 1998' name='created'/>
    <meta content='20 December 2002' name='imported'/>
    <meta content='26 February 2008' name='modified'/>
    <meta content='KelpTank.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/IntegerSequencerPumpHouse.x3d' name='ident
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='Click pump house for next cone' orientation='-0.969 0.239 0.056 0.13' position='1.66 1.34 5.95'/>
    <Group DEF='PumpGeometry'>
      <Transform>
      <Group>
    </Group>
    <Transform translation='3 2 0.5'>
      <Transform DEF='ConeRotationTransform'>
        <OrientationInterpolator DEF='Rotator' key='0 .33 .67 1' keyValue='0 0 1 0 0 0 1 2 0 0 1 4 0 0
        <ROUTE fromField='value_changed' fromNode='Rotator' toField='set_rotation' toNode='ConeRotatio
        <TimeSensor DEF='Timer' cycleInterval='2' enabled='true' loop='true' startTime='1'/>
        <ROUTE fromField='fraction_changed' fromNode='Timer' toField='set_fraction' toNode='Rotator'/>
        <!-- positioning ROUTEs immediately following animation nodes helps keep track of event logic
        <Switch DEF='Switcher' whichChoice='0'>
          <Transform>
          <Transform>
          <Transform>
        </Switch>
        <IntegerSequencer DEF="Sequencer" key="0 .33 .67 1" keyValue="0 1 2 -1" />
        <ROUTE fromField="value_changed" fromNode="Sequencer" toField="whichChoice" toNode="Switcher"
        <!-- sending true event (on TouchSensor select) advances to next value, sending false event (on TouchSensor deselect) has no effect -->
        <ROUTE fromField="isActive" fromNode="RunPump" toField="next" toNode="Sequencer" />
      </Transform>
    </Transform>
    <Transform scale='0.5 0.5 0.5' translation='1.0 1.1 -1.5'>
```

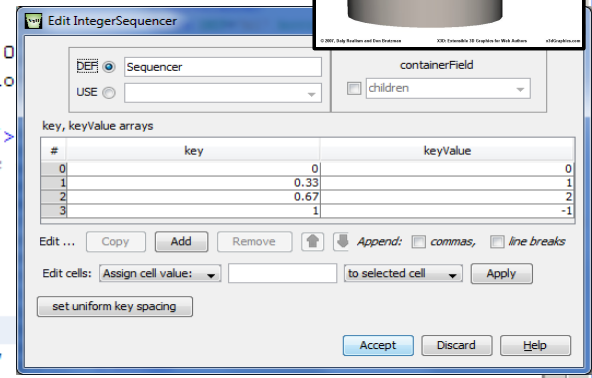
whichChoice

value_changed

next

TouchSensor.isActive





TouchSensor drives IntegerSequencer outputs to provide clickable stream of integer values

76:26 INS

IntegerSequencerRoadSignSwitcher.x3d - Editor

IntegerSequencerRoadSignSwitcher.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.2//EN" "http://www.web3d.org/specifications/x3d-3.2.dtd">
<X3D profile='Interactive' version='3.2' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/sp
  <head>
    <meta content='IntegerSequencerRoadSignSwitcher.x3d' name='title'/>
    <meta content='Switch among different road signs using IntegerSequencer' name='description'/>
    <meta content='Don Brutzman' name='creator'/>
    <meta content='2 January 2009' name='created'/>
    <meta content='2 January 2009' name='modified'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter07-EventAnim
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='X3D-Edit 3.2, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../../license.html' name='license'/>
  </head>
  <Scene>
    <Background groundColor='0.2 0.2 0.2' skyColor='0.2 0.2 0.2'/>
    <Viewpoint description='Road sign switcher, 6m away' position='0 0 6'/>
    <Shape DEF='Frame'>
      <!-- Box framee scaled to be wider, taller and thinner than Box holding image -->
      <Box size='3.5 2.7 0.05'/>
      <Appearance>
    </Shape>
    <!-- Modify the whichChoice value in this Switch to 0, 1, 2 or 3 to see the various
    <Switch DEF='Switch' whichChoice='0'>
      <!-- whichChoice values are 0, 1, 2, 3 for these four children -->
      <Shape>
        <!-- Box size scaled to match aspect ratio of original images -->
        <Box DEF='SignHolder' size='3.072 2.304 0.1'/>
        <Appearance>
          <!-- each image size reduced from 1-1.5M down to 150-200KB using GIMP for faster downloading, quality still looks good -->
          <ImageTexture url='"RoadSignRoadWorkAhead.jpg" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter07-EventAnimationInterpolation/Roa
        </Appearance>
      </Shape>
      <Shape>
        <Box USE='SignHolder'/>
        <Appearance>
          <ImageTexture url='"RoadSignExpectDelays.jpg" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter07-EventAnimationInterpolation/Roads
        </Appearance>
      </Shape>
      <Shape>
      <Shape>
    </Switch>
    <IntegerSequencer DEF='Sequencer' key='0 0.25 0.5 0.75 1' keyValue='0 1 2 3 0'/>
    <TimeSensor DEF='Time' cycleInterval='8' enabled='true' loop='true'/>
    <ROUTE fromField='value_changed' fromNode='Sequencer' toField='whichChoice' toNode='Switch'/>
    <ROUTE fromField='fraction_changed' fromNode='Time' toField='set_fraction' toNode='Sequencer'/>
```

whichChoice

value_changed

TimeSensor drives IntegerSequencer outputs
to provide steady stream of integer values

21:1 INS

IntegerSequencer	**IntegerSequencer generates periodic discrete Integer values that can be ROUTEd to other Integer attributes. Typical input: ROUTE someTimeSensor.fraction_changed TO someInterpolator.set_fraction Typical output: ROUTE someInterpolator.value_changed TO destinationNode.set_attribute.**
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
key	**[key: accessType inputOutput, type MFFloat CDATA #IMPLIED]** Definition parameters for linear-interpolation function time intervals, in increasing order and corresponding to keyValues. **Hint:** number of keys must match number of keyValues!
keyValue	**[keyValue: accessType inputOutput, type MFInt32 CDATA #IMPLIED]** Output values for linear interopolation, each corresponding to time-fraction keys. **Hint:** number of keys must match number of keyValues!
set_fraction	**[set_fraction: inputOnly type SFFloat CDATA #FIXED ""]** set_fraction selects input key for corresponding keyValue output.
value_changed	**[value_changed: accessType outputOnly, type SFInt32 CDATA#FIXED ""]** Single intermittent output value determined by current key time and corresponding keyValue pair.
previous	**[previous: accessType inputOnly, type SFBool (true|false) "0"]** Trigger previous output value in keyValue array. **Hint:** loops from first to last if necessary.
next	**[next: accessType inputOnly, type SFBool (true|false) "0"]** Trigger next output value in keyValue array. **Hint:** loops from last to first if necessary.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

IntegerTrigger node

IntegerTrigger converts input SFBool *set_boolean* events into output SFInt32 *triggerValue* events

- Convenient type conversion

Only one integer output value possible

- Saved in field *integerKey*
- If more than one *integerKey* is needed, then create a new IntegerTrigger node for each condition

Typical target for output:

- specific Switch *whichChoice* child geometry

IntegerTrigger.x3d - Editor

IntegerTrigger.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specif
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSch
  <head>
    <meta content='IntegerTrigger.x3d' name='title'/>
    <meta content='An IntegerTrigger node controls the display of colored text.
    <meta content='Don Brutzman and Leonard Daly' name='authors'/>
    <meta content='1 March 2008' name='created'/>
    <meta content='2 March 2008' name='modified'/>
    <meta content='KelpTank.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/IntegerTrigger.x3d' name='identifier'/>
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='Select text to change color' position='0 0 6'/>
    <Switch DEF='SwitchColoredTextGroups' whichChoice='0'>
      <Group>
        <Shape>
          <Appearance>
          <Text DEF='MessageText' string='Select text to change color'>
            <FontStyle justify='"MIDDLE" "MIDDLE"'/>
          </Text>
        </Shape>
        <!-- TransparentBox makes selection of Text using TouchSensor easier for user -->
        <Shape>
        <TouchSensor DEF='Touch1' description='touch to activate'/>
      </Group>
      <!-- Reuse Text geoemetry, paired with different Material colors -->
      <Group>
      <Group>
    </Switch>
    <!-- IntegerTrigger nodes used to switch child shapes, must use separate one for each output value -->
    <IntegerTrigger DEF='Filter1' integerKey='1'/>
    <IntegerTrigger DEF='Filter2' integerKey='2'/>
    <IntegerTrigger DEF='Filter3' integerKey='0'/>
    <ROUTE fromField='isActive' fromNode='Touch1' toField='set_boolean' toNode='Filter1'/>
    <ROUTE fromField='isActive' fromNode='Touch2' toField='set_boolean' toNode='Filter2'/>
    <ROUTE fromField='isActive' fromNode='Touch3' toField='set_boolean' toNode='Filter3'/>
    <ROUTE fromField='triggerValue' fromNode='Filter1' toField='whichChoice' toNode='SwitchColoredTextGroups'/>
    <ROUTE fromField='triggerValue' fromNode='Filter2' toField='whichChoice' toNode='SwitchColoredTextGroups'/>
    <ROUTE fromField='triggerValue' fromNode='Filter3' toField='whichChoice' toNode='SwitchColoredTextGroups'/>
  </Scene>
</X3D>
```

whichChoice

isActive

set_boolean

triggerValue

63:1 INS

Edit IntegerTrigger

DEF Filter1

USE Filter2

containerField

children

integerKey 1

IntegerTrigger converts boolean true or time input events to integer output

Accept Discard Help

Switch node selects correct message:

Select text to change color

Select text to change color

Select text to change color

Each of 3 TouchSensors activates matching IntegerTrigger to provide integer value for corresponding child of Switch

IntegerTrigger	IntegerTrigger converts boolean true or time input events to integer value (suitable for Switch node).
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
set_boolean	**[set_boolean: accessType inputOnly, type SFBool (true|false) #FIXED ""]** If set_boolean input is true, trigger output of integer value.
integerKey	**[integerKey: accessType inputOutput, type SFInt32 CDATA #FIXED "-1"]** integerKey is value for output when triggered.
triggerValue	**[triggerValue: accessType outputOnly, type SFInt32 CDATA #FIXED ""]** triggerValue provides integer event output matching integerKey when true set_boolean received.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

TimeTrigger node

TimeTrigger converts input SFBool *set_boolean* events into output SFTime *triggerTime* events

- Convenient type conversion
- Input value can be either *true* or *false*
  - So if you don't want *false* values to produce a time event, include BooleanFilter beforehand to filter out *false* values
  - Same principle: hook up correct combinations as needed
- Output value matches current clock time

Typical targets for output:

- Starting, stopping, pausing, or resuming either a TimeSensor, MovieTexture or AudioClip node

TimeTriggerTest.x3d - Editor

TimeTriggerTest.x3d

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3  <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specificat
4    <head>
5      <meta content='TimeTriggerTest.x3d' name='title'/>
6      <meta content='Test of TimeTrigger node.' name='description'/>
7      <meta content='8 October 2007' name='created'/>
8      <meta content='28 September 2008' name='modified'/>
9      <meta content='Leonard Daly and Don Brutzman' name='authors'/>
10     <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/TimeTriggerTest.x3d' name='identifier'/>
11     <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
12     <meta content='../license.html' name='license'/>
13   </head>
14   <Scene>
15     <Background skyColor='1 1 1'/>
16     <Viewpoint description='TimeTrigger test' position='0 0 7'/>
17     <Transform>
18       <Shape>
19         <Appearance>
20           <Material diffuseColor='.6 0 .8'/>
21         </Appearance>
22         <Text string='"Touch text to print" "output values on console"'>
23           <FontStyle justify='"MIDDLE" "MIDDLE"'/>
24         </Text>
25       </Shape>
26       <TouchSensor DEF='TextTouched' description='touch to activate'/>
27     </Transform>
28     <BooleanFilter DEF='Filter'/>
29     <ROUTE fromField='isActive' fromNode='TextTouched' toField='set_boolean' toNode='Filter'/>
30     <TimeTrigger DEF='Trigger'/>
31     <ROUTE fromField='inputTrue' fromNode='Filter' toField='set_boolean' toNode='Trigger'/>
32     <Script DEF='TraceScript'>
33       <field accessType='inputOnly' name='printBoolean' type='SFBool'/>
34       <field accessType='inputOnly' name='printTimestamp' type='SFTime'/>
35 <![CDATA[
36 ecmascript:
37     function printBoolean (value) {
38       Browser.print ('TouchSensor isActive="' + value + '"\n');
39     }
40     function printTimestamp (value) {
41       Browser.print ('Trigger triggerTime="' + value + '"\n');
42     }
43 ]]>
44     </Script>
45     <ROUTE fromField='triggerTime' fromNode='Trigger' toField='printTimestamp' toNode='TraceScript'/>
46     <ROUTE fromField='isActive' fromNode='TextTouched' toField='printBoolean' toNode='TraceScript'/>
47   </Scene>
48 </X3D>
```

isActive

isActive (unfiltered)

Touch text to print output values on console

Instant Player Console

File Window ?

```
Trigger triggerTime="179815.259"
TouchSensor isActive="true"
TouchSensor isActive="false"
Trigger triggerTime="179827.04"
TouchSensor isActive="true"
TouchSensor isActive="false"
```

Edit TimeTrigger

DEF Trigger

USE

containerField

children

TimeTrigger converts boolean true events to time events

Accept Discard Help

TimeTrigger converts boolean input event into SFTime output event

30:17 INS

TimeTrigger	TimeTrigger converts boolean true events to time events.
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
set_boolean	**[set_boolean: accessType inputOnly, type SFBool (true|false) #FIXED ""]** If set_boolean input is true, trigger output time value.
triggerTime	**[triggerTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** triggerTime is output time event, sent when set_boolean input is true.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

Script node

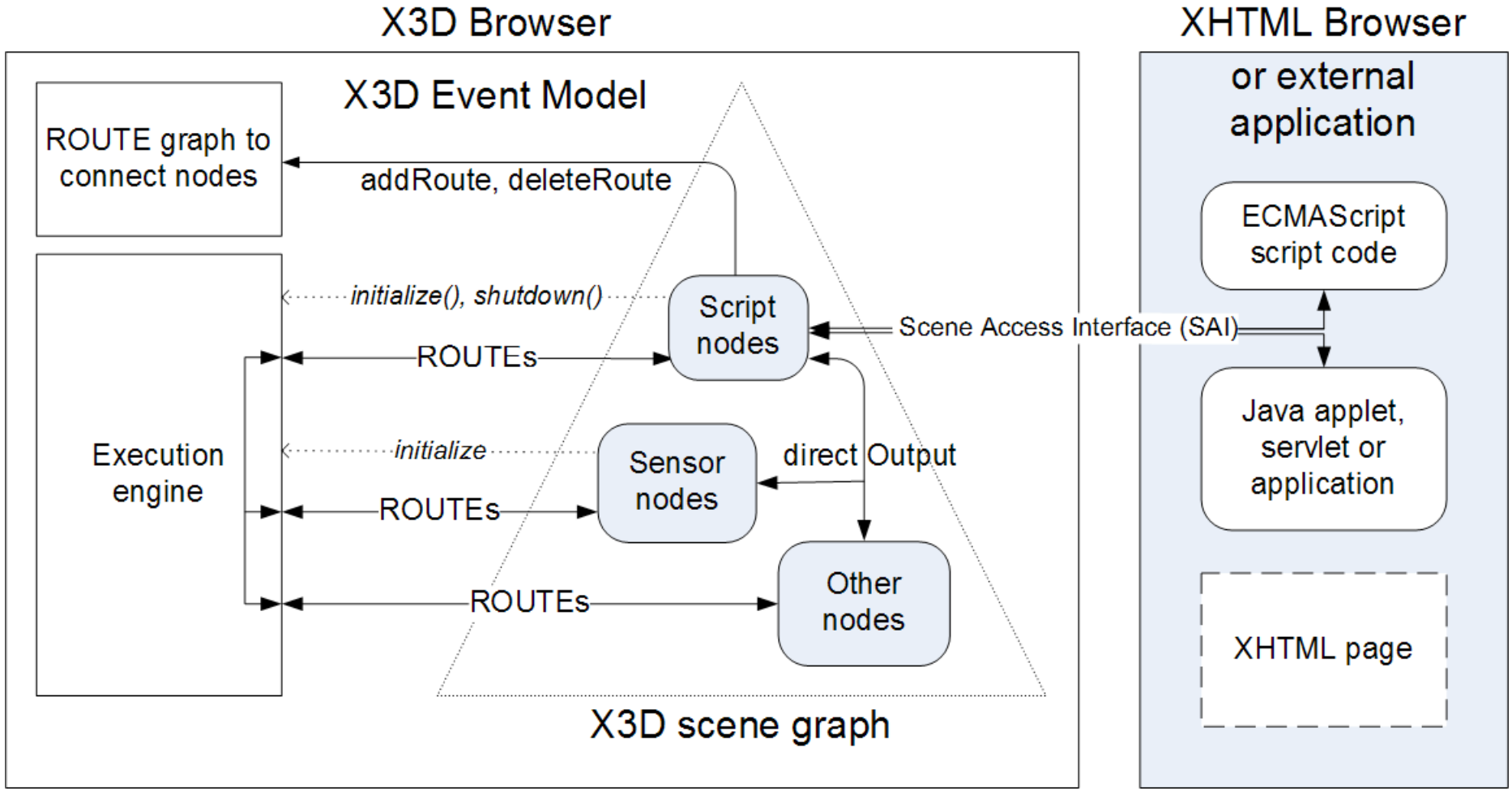
Script encapsulates programmatic source code

- Execution is triggered via arrival of input event
  - Each input field (i.e. method) receives a value, ← Declarative
  - performs algorithmic calculations, then ← Imperative
  - can return response event(s) via output fields ← Declarative

This approach preserves X3D event model for animation, allowing Script code within the scene graph to perform complex computations

- In response to ROUTEd scene-graph event inputs
- Similarly can return typed X3D values for use in scene graph via ROUTE connections

Event model relationships



Simple X3D execution model

- Display image from double buffer
- Advance and freeze the simulation clock
- Generate and process all events
  - Then stop, don't loop indefinitely!
- Draw new image in back buffer
- Swap buffer, replace with new image

X3D execution model

0. Prior image frame is already displayed on screen
1. Update camera based on currently bound Viewpoint
2. Evaluate inputs from sensors and event producers, then place events in queue for delivery
3. ROUTE events to defined destinations, updating scene-graph fields as appropriate
4. If any events generated as a result of steps 2 & 3, repeat until all events delivered in event cascade
5. Pixels in next screen update are rendered to image
6. Double-buffer display swaps new, old frame images
7. Browser's simulation clock is updated to match system's real-time clock (advancing 'one tick')
8. Single loop done, repeat from step 1 indefinitely

Event model and event cascade

Script operation must commence and complete in between scene-graph rendering cycles

- Input events can provoke output events
- Output events may in turn provoke release of other events, known as ***event cascade***
- Scene graph values updated as direct result

Event loops not allowed

- Repeated values arriving at input fields are ignored
- Infinite loops thus prevented

Next frame drawn when event cascade complete

field declarations

Each Script can contain <field> definitions

- Which are the interface for the Script
- Zero or more <field> definitions allowed

<field> definitions must be exactly correct

- Define *name*, *type*, *accessType*, and initial *value*
- SFNode, MFNode initializations are contained
- *accessType* initializeOnly, inputOnly must have an initial *value*
- inputOnly, outputOnly have no initial *value*

Field-Type Names	Description	Default Values
SFBool	Single-Field boolean value	false (XML syntax) or FALSE (ClassicVRML syntax)
MFBool	Multiple-Field boolean array	Empty list
SFColor	Single-Field color value, RGB	0 0 0
MFColor	Multiple-Field color array, RGB	Empty list
SFColorRGBA	Single-Field color value, red-green-blue alpha (opacity)	0 0 0 0
MFColorRGBA	Multiple-Field color array, red-green-blue alpha (opacity)	Empty list
SFInt32	Single-Field 32-bit integer value	0
MFInt32	Multiple-Field 32-bit integer array	Empty list
SFFloat	Single-Field single-precision floating-point value	0.0
MFFloat	Multiple-Field single-precision floating-point array	Empty list
SFDouble	Single-Field double-precision floating-point value	0.0
MFDouble	Multiple-Field double-precision array	Empty list
SFImage	Single-Field image value	0 0 0 Contains special pixel-encoding values, see Chapter 5 for details

MFImage	Multiple-Field image value	Empty list
SFNode	Single-Field node	Empty node, NULL
MFNode	Multiple-Field node array of peers	Empty list
SFRotation	Single-Field rotation value using 3-tuple axis, radian-angle form	0 0 1 0
MFRotation	Multiple-Field rotation array	Empty list
SFString	Single-Field string value	Empty string, representable as two adjacent quotation marks
MFString	Multiple-Field string array	Empty list
SFTime	Single-Field time value	–1, sentinel indicating no time value.
MFTime	Multiple-Field time array	Empty list
SFVec2f/SFVec2d	Single-Field 2-float/2-double vector value	0 0
MFVec2f/MFVec2d	Multiple-Field 2-float/2-double vector array	Empty list
SFVec3f/SFVec3d	Single-Field vector value of 3-float/3-double values	0 0 0
MFVec3f/MFVec3d	Multiple-Field vector array of 3-float/3-double values	Empty list

field naming conventions by accessType

- Script **inputOnly** fields become method names, with event value passed as first parameter
  - Event timestamp optionally passed as second parameter
- Script **outputOnly** fields are set via assignment statements (i.e. on left-hand side of equal sign)
- Script **initializeOnly** fields can be used internally, and also reset to different values to save state
  - But cannot receive values via external ROUTE connection
- Script **inputOutput** fields are for function receiving events, or saving state, or sending output values
  - But naming can be confusing, and so inputOutput fields are usually best avoided (TODO add link once open)

Script node life cycle

Begin loading X3D scene, including Inline and ExternPrototype nodes

`function initialize()` method invoked

X3D scene load complete, begin scene rendering and animation

Event generated in scene, ROUTEd to Script method which matches input event name

```
<Script>
  <field name="triggerField" type="SFString" accessType="inputOnly">
  <field name="acknowledged" type="SFBool"   accessType="outputOnly">
<![CDATA[
ecmascript:

function triggerField (value, timestamp)
{
  Browser.println ('triggerField event with value=' + value +
  ' received at time=' + timestamp);
  acknowledged = true;
}
function initialize ()   // optional
{
  // perform setups here
}
function shutdown () { } // optional
]></Script>
```

Queued output event `acknowledged` sent via ROUTE to scene graph

Repeat invocation of Script methods until complete, break repeating loops if necessary
Scene rendering remains suspended until scene animation event cascade complete

Event cascade complete: render scene output, update timestamp

Continue event-based animation and output rendering until X3D browser halted

`function shutdown()` method invoked

ECMAScript (JavaScript)

ECMAScript is primary language for X3D scripting

- Required for X3D players that support scripting
- Originally (and often) called JavaScript
- Renamed ECMAScript when standardized via European Computer Manufacturers Association http://www.ecma-international.org
- Scene Access Interface (SAI) defines standard interfaces for interaction with scene graph

Can be embedded within X3D scene, or kept separately in an external .js file

- X3D-Edit provides code colorization and syntax support when editing separate .js file

Java

Java classes can be used for X3D scripting

- Optional for X3D players that support scripting
- Scene Access Interface (SAI) defines standard interfaces for interaction with scene graph

Has some advantages over ECMAScript

- Network and database connectivity
- Large set of class libraries & functionality available

Java script code not covered in this book chapter

- Xj3D has tutorials on use of Java with X3D online at http://www.xj3d.org/tutorials

Java generation example

Java can also be used for creation of X3D scenes

- Output from standalone programs
- Output from existing programs

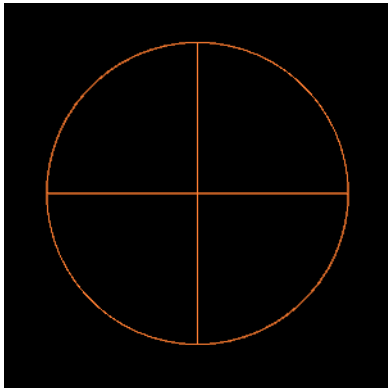
Example program: CircleLines.java

- Invoked via command line
- User specifies number of points in circle (default 24)

```
Usage:   java CircleLines [numberOfSegments] [ > CircleLinesFileName.x3d ]
Example: java CircleLines 60 > CircleLines60.x3d
```

Example scene results:

- CircleLinesExample.x3d
- CircleLinesExample60.x3d



url field and CDATA block

Local and online url addresses for script code contained as elements of url string array

- Essentially same approach as other url fields
- Each address intended to point to a copy of same functionally equivalent scripting code

One url value can also contain `ecmascript:` code

- This approach is used in ClassicVRML encoding

Preferred X3D approach for XML encoding is to encapsulate in CDATA block, which prevents unintended XML parsing of character data

CDATA character data => “hey XML leave it alone!!”

CDATA block syntax

Text characters within character data (CDATA) blocks are preserved verbatim and do not undergo XML parsing.

```
<Script>
    <field name="a" type="SFBool"  accessType="initializeOnly" value="true"/>
    <field name="b" type="SFInt32" accessType="inputOutput"    value="1"/>
    <field name="c" type="SFFloat" accessType="inputOnly"/>
    <field name="c" type="SFColor" accessType="outputOnly"/>
<![CDATA[
ecmascript:

// see newECMAscript.js for example javascript code for your Script
]]>
  </Script>
```

newECMAscript.js - Editor

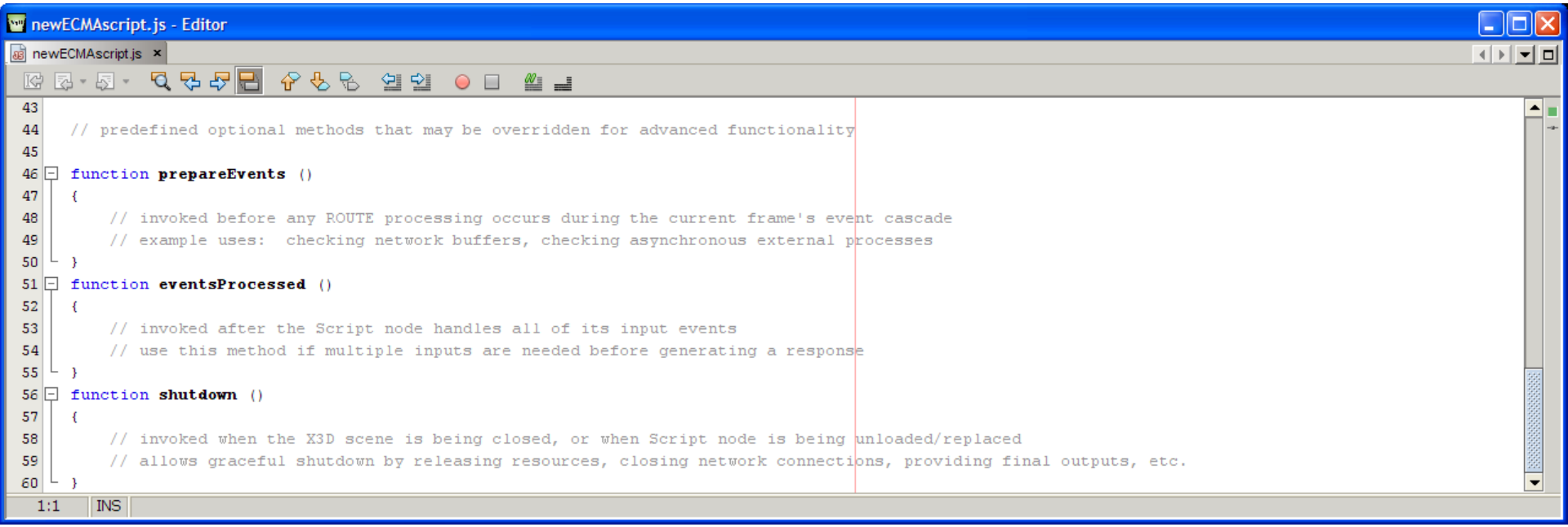
newECMAscript.js

```
// ecmascript:  // uncomment this line if this code is copied into a CDATA section, embedded inside an X3D Script node

// Author TODO  * authors please update metadata and information entries in this file *

// Description: Editable example ECMAscript (javascript) source file for use with X3D Script node
// Filename:    newECMAscript.js
// Author:      Don Brutzman
// Identifier:  http://www.web3d.org/x3d/content/examples/newECMAscript.js
// Created:     11 December 2007
// Revised:     16 August   2008
// Reference:   http://www.web3d.org/x3d/content/X3dTooltips.html#Script
// Reference:   http://www.web3d.org/x3d/content/examples/X3dSceneAuthoringHints.html#Scripts
// Reference:   http://www.web3d.org/x3d/specifications/ISO-IEC-FDIS-19775-1.2-X3D-AbstractSpecification/Part01/components/scripting.html
// Reference:   http://www.web3d.org/x3d/specifications/ISO-IEC-19777-1-X3DLanguageBindings-ECMAScript/Part1/X3D_ECMAScript.html
// License:     ../../license.html

function initialize ()
{
    // The initialize() function is automatically invoked when the Script node is first activated, prior to other events

    var scriptName = 'newECMAscript.js'; // Author TODO:  authors need to edit this scriptName or else remove all uses

    Browser.println ('[' + scriptName + '] ' + 'initialize() commmenced...');

    // TODO:  authors can insert initialization code here (if any is needed)

    Browser.println ('[' + scriptName + '] ' + 'initialize() complete');
    // initialize() can also be invoked by other functions, if appropriate
}

// inputOnly fields can only appear as names of handling functions (like the following)
// failure to provide a named method for each inputOnly field means Script ignores events at run time

function myInputOnlyFieldName (myInputValue, timestamp) // all timestamp parameters optional
{
    // Author TODO:  authors can insert script code here; variable names must match Script field definitions
    //               outputOnly      fields can only be on left-hand side  (LHS) of assignment statements
    //               initializeOnly  fields can only be on right-hand side (RHS) of assignment statements
    //               inputOutput     fields can either be on LHS or RHS of assignment statements

    myOutputOnlyFieldName = someExpression (myInputValue);  // Author TODO:  replace this assignment statement
}

// predefined optional methods that may be overridden for advanced functionality

function prepareEvents ()
{
```

6:33 INS

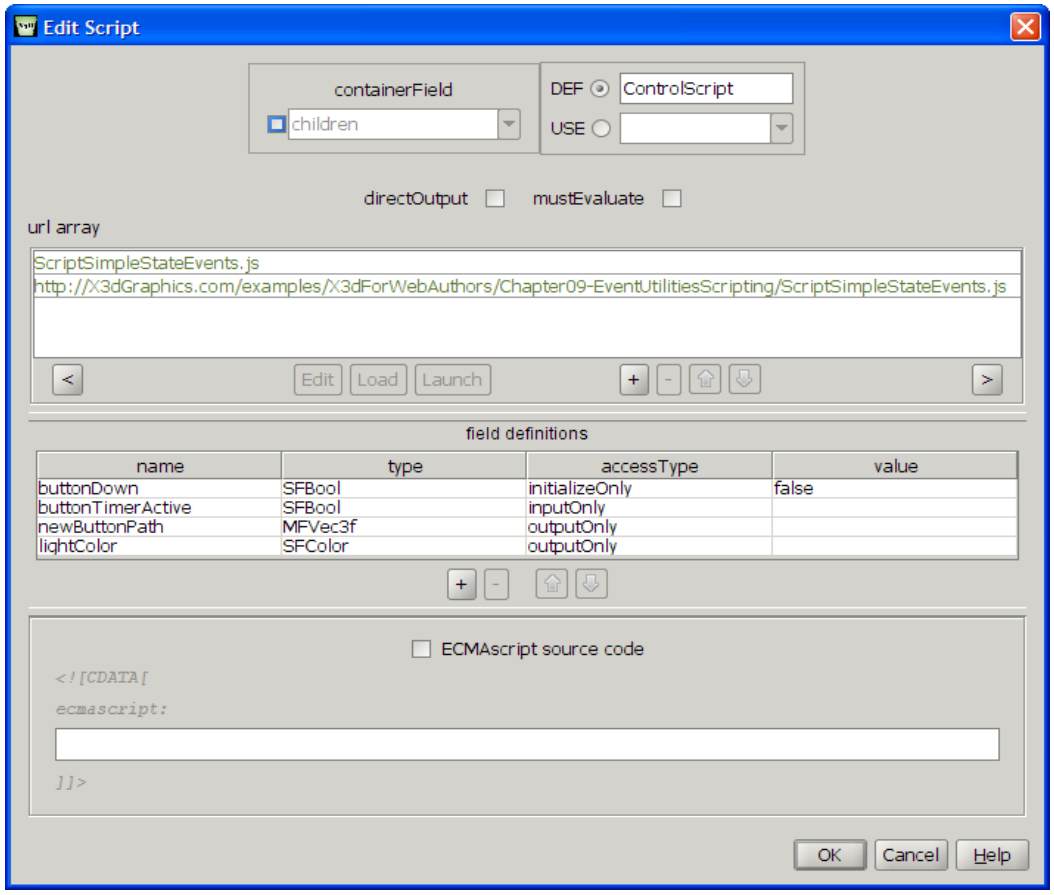
newECMAscript.js part 2

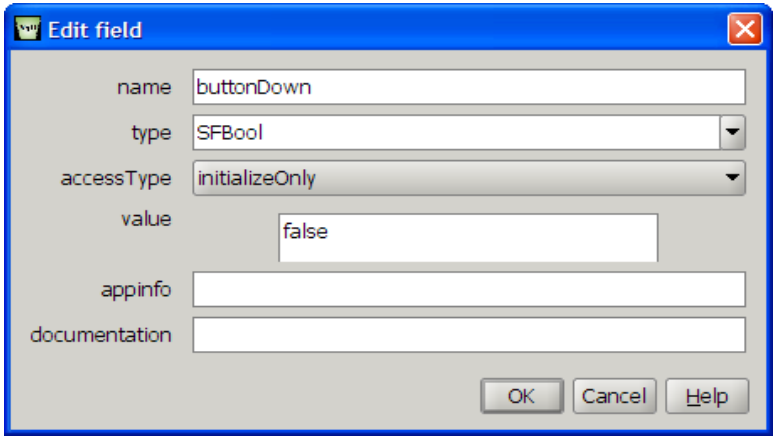
... continued from previous slide:

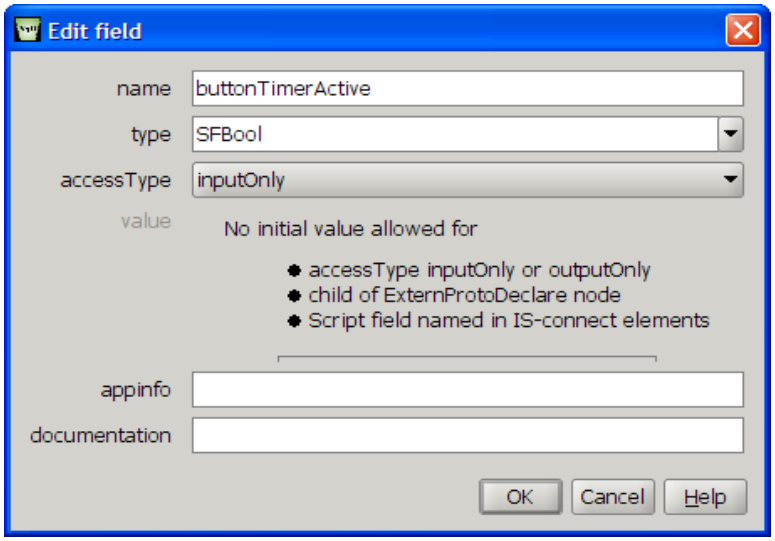


Worth noting: these methods are rarely used

<Script> and <field> editors







Example: ScriptSimpleStateEvents.x3d

Pushbutton switch changes lamp color

- TouchSensor detects button select, acts as trigger
- TimeSensor animates pushbutton cylinder
- Script invoked, checks current state to decide logic
  - If button now down, color lamp yellow and reset path
  - If button now up, color lamp grey and reset path

Script code for this example kept in external file

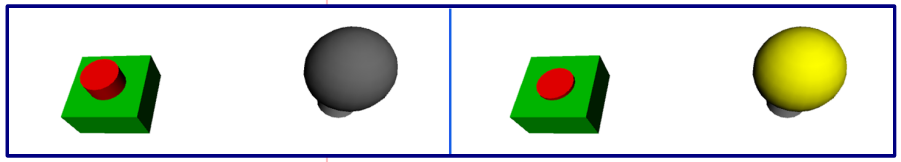
- Still has access to all defined fields from within code
- Don't include `ecmascript:` header in external file

X3D event model still governs animation flow

ScriptSimpleStateEvents.x3d - Editor

ScriptSimpleStateEvents.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org
  <head>
    <meta content='ScriptSimpleStateEvents.x3d' name='title'/>
    <meta content='A Script node drives an animated push button that turns a light on and off.' name='description'/>
    <meta content='Leonard Daly and Don Brutzman' name='creator'/>
    <meta content='10 June 2006' name='created'/>
    <meta content='1 March 2008' name='modified'/>
    <meta content='http://X3dGraphics.com' name='reference'/>
    <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
    <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.x3d' name='identi
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='click switch to toggle light' orientation='-0.993 0.101 -0.063 1.06' position='0.06 3.63 2.29'/>
    <Transform DEF='PushBox' translation='-2 0 0'>
      <Transform DEF='ControlBox'>
        <Shape>
          <Appearance>
            <Material diffuseColor='0 .8 0'/>
          </Appearance>
          <Box size='1 .5 1'/>
        </Shape>
      </Transform>
      <Transform DEF='ControlButton' translation='0 .25 0'>
        <TouchSensor DEF='ButtonTouch' description='touch to toggle'/>
        <Shape>
          <Appearance>
            <Material diffuseColor='1 0 0'/>
          </Appearance>
          <Cylinder DEF='Button' bottom='false' height='.5' radius='.25'/>
        </Shape>
        <PositionInterpolator DEF='ButtonMover' key='0 1' keyValue='0 .25 0 0 .05 0'/>
        <TimeSensor DEF='ButtonTimer' cycleInterval='1' enabled='true' loop='false'/>
        <ROUTE fromField='touchTime' fromNode='ButtonTouch' toField='startTime' toNode='ButtonTimer'/>
        <ROUTE fromField='fraction_changed' fromNode='ButtonTimer' toField='set_fraction' toNode='ButtonMover'/>
        <ROUTE fromField='value_changed' fromNode='ButtonMover' toField='translation' toNode='ControlButton'/>
      </Transform>
    </Transform>
```



touch to start

Timer isActive

Button position animated

new button path setup for next time

31:21 INS

ScriptSimpleStateEvents.x3d - Editor

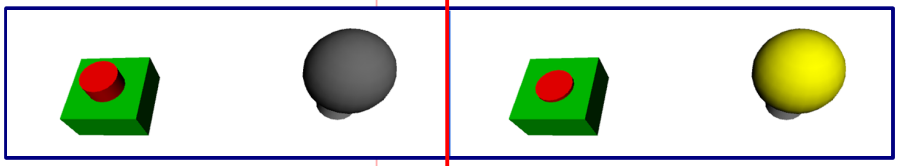
ScriptSimpleStateEvents.x3d

```
    <Transform DEF='Lamp' translation='1 0 0'>
      <Transform DEF='Base'>
        <Shape>
          <Appearance>
            <Material/>
          </Appearance>
          <Cylinder height='.5' radius='.25'/>
        </Shape>
      </Transform>
      <Transform DEF='Bulb' translation='0 .5 0'>
        <Shape>
          <Appearance>
            <Material DEF='LightBulbMaterial' diffuseColor='.4 .4 .4'/>
          </Appearance>
          <Sphere radius='.5'/>
        </Shape>
      </Transform>
    </Transform>
    <Script DEF='ControlScript' url='"ScriptSimpleStateEvents.js" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScr
      <field accessType='initializeOnly' name='buttonDown' type='SFBool' value='false'/>
      <field accessType='inputOnly' name='buttonTimerActive' type='SFBool'/>
      <field accessType='outputOnly' name='newButtonPath' type='MFVec3f'/>
      <field accessType='outputOnly' name='lightColor' type='SFColor'/>
    </Script>
    <ROUTE fromField='isActive' fromNode='ButtonTimer' toField='buttonTimerActive' toNode='ControlScript'/>
    <ROUTE fromField='newButtonPath' fromNode='ControlScript' toField='keyValue' toNode='ButtonMover'/>
    <ROUTE fromField='lightColor' fromNode='ControlScript' toField='diffuseColor' toNode='LightBulbMaterial'/>
  </Scene>
</X3D>
```

Timer isActive

ButtonMover keyValue array

PositionInterpolator gets reversed



LightBulbMaterial color gets swapped

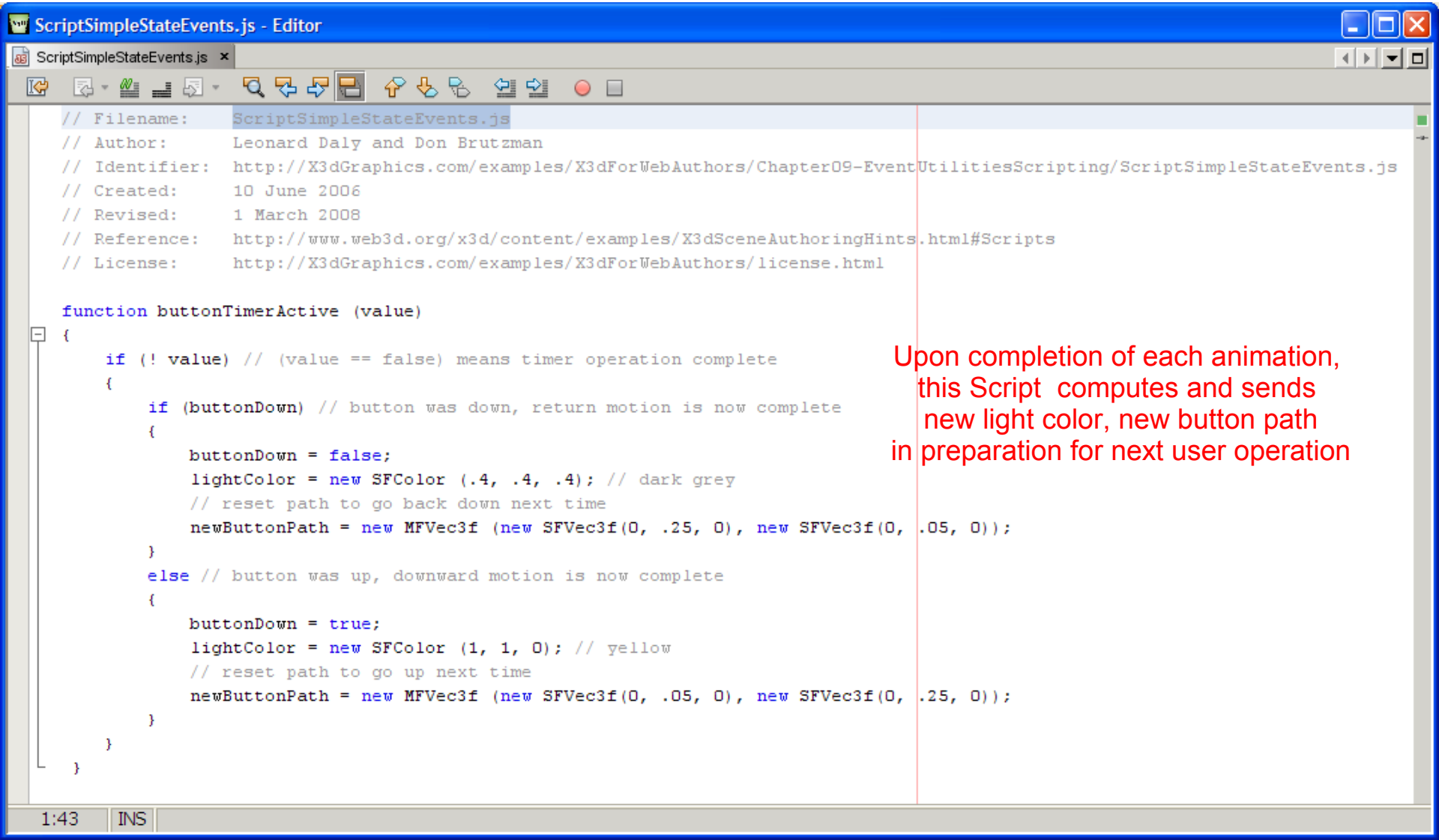
Script computes and sends new light color, new button path...

31:21 INS

Script located in external ScriptSimpleStateEvents.js file

- buttonTimerActive function name matches inputOnly event; value contains passed event
- outputOnly events set by assignment statements

ScriptSimpleStateEvents.js



Example: ScriptEvents.x3d

Clicking pump house starts cone animation

- TouchSensor detects pump select, acts as trigger
- TimeSensor drives Script response
- Script invoked, computes rotation and translations

Script code for this example kept within .x3d file

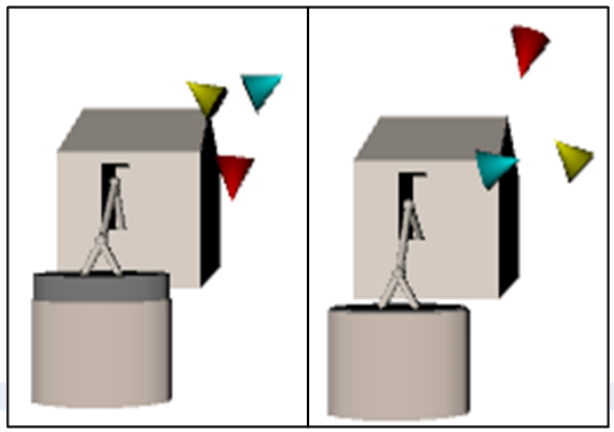
- Still has access to all defined fields from within code
- Must include `ecmascript:` header when internal
- CDATA block ensures that original text is preserved

X3D event model still governs animation flow

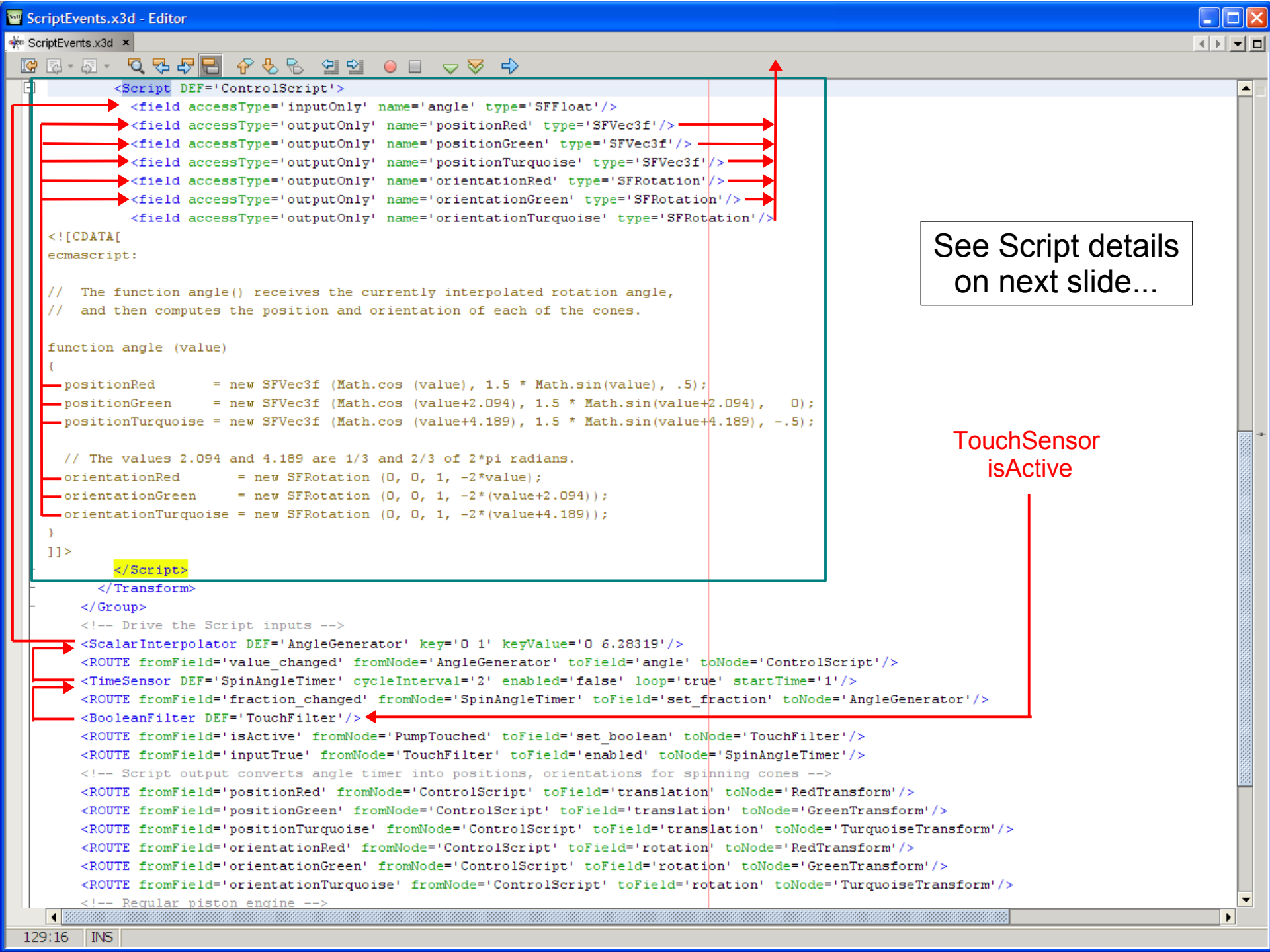
ScriptEvents.x3d - Editor

ScriptEvents.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org
  <head>
    <meta content='ScriptEvents.x3d' name='title'/>
    <meta content='A Script node drives the position and orientation of orbiting cones near the pump house.' name='description'/>
    <meta content='Todd Gagnon and Mark A. Boyd' name='authors'/>
    <meta content='Xeena VRML importer, X3D-Edit 3.1, http://www.web3d.org/x3d/content/README.X3D-Edit.html' name='translator'/>
    <meta content='8 June 1998' name='created'/>
    <meta content='20 December 2002' name='imported'/>
    <meta content='2 March 2008' name='modified'/>
    <meta content='KelpTank.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptEvents.x3d' name='identifier'/>
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='click pump house to spin cones' position='3 1 7'/>
    <Viewpoint description='Book View' orientation='-0.969 0.239 0.056 0.13' position='1.66 1.34 5.95'/>
    <Group>
    <Group>
      <Transform scale='0.5 0.5 0.5' translation='1.0 1.1 -1.5'>
        <Transform>
        <TimeSensor DEF='PistonClock' cycleInterval='5.5' enabled='true' loop='true'/>
        <PositionInterpolator DEF='PistonPath' key='0.0 0.3 0.32 0.5 0.75 1.0' keyValue='-0.4 -2.3 4.0 -0.4 -1.5 4.0 -0.4 -1.5 4.0 -0.4 -2.3 4.0
        <Transform translation='3 2 4'>
          <Transform DEF='ThreeCones'>
            <Transform DEF='RedTransform' rotation='0 0 1 3.14' translation='0 -1.5 .5'>
              <Shape>
                <Appearance>
                  <Material diffuseColor='1 0 0'/>
                </Appearance>
                <Cone DEF='b1' bottomRadius='.5' height='1'/>
              </Shape>
            </Transform>
            <Transform>
            <Transform>
          </Transform>
        </Transform>
        <Script DEF='ControlScript'>
          <field accessType='inputOnly' name='angle' type='SFFloat'/>
          <field accessType='outputOnly' name='positionRed' type='SFVec3f'/>
```



129:16 INS


ScriptEvents.x3d - Editor
ScriptEvents.x3d
<Script DEF='ControlScript'>
<field accessType='inputOnly' name='angle' type='SFFloat'/>
<field accessType='outputOnly' name='positionRed' type='SFVec3f'/>
<field accessType='outputOnly' name='positionGreen' type='SFVec3f'/>
<field accessType='outputOnly' name='positionTurquoise' type='SFVec3f'/>
<field accessType='outputOnly' name='orientationRed' type='SFRotation'/>
<field accessType='outputOnly' name='orientationGreen' type='SFRotation'/>
<field accessType='outputOnly' name='orientationTurquoise' type='SFRotation'/>
<![CDATA[
ecmascript:
// The function angle() receives the currently interpolated rotation angle,
// and then computes the position and orientation of each of the cones.
function angle (value)
{
positionRed = new SFVec3f (Math.cos (value), 1.5 * Math.sin(value), .5);
positionGreen = new SFVec3f (Math.cos (value+2.094), 1.5 * Math.sin(value+2.094), 0);
positionTurquoise = new SFVec3f (Math.cos (value+4.189), 1.5 * Math.sin(value+4.189), -.5);
// The values 2.094 and 4.189 are 1/3 and 2/3 of 2*pi radians.
orientationRed = new SFRotation (0, 0, 1, -2*value);
orientationGreen = new SFRotation (0, 0, 1, -2*(value+2.094));
orientationTurquoise = new SFRotation (0, 0, 1, -2*(value+4.189));
}
]]>
</Script>
</Transform>
</Group>
<!-- Drive the Script inputs -->
<ScalarInterpolator DEF='AngleGenerator' key='0 1' keyValue='0 6.28319'/>
<ROUTE fromField='value_changed' fromNode='AngleGenerator' toField='angle' toNode='ControlScript'/>
<TimeSensor DEF='SpinAngleTimer' cycleInterval='2' enabled='false' loop='true' startTime='1'/>
<ROUTE fromField='fraction_changed' fromNode='SpinAngleTimer' toField='set_fraction' toNode='AngleGenerator'/>
<BooleanFilter DEF='TouchFilter'/>
<ROUTE fromField='isActive' fromNode='PumpTouched' toField='set_boolean' toNode='TouchFilter'/>
<ROUTE fromField='inputTrue' fromNode='TouchFilter' toField='enabled' toNode='SpinAngleTimer'/>
<!-- Script output converts angle timer into positions, orientations for spinning cones -->
<ROUTE fromField='positionRed' fromNode='ControlScript' toField='translation' toNode='RedTransform'/>
<ROUTE fromField='positionGreen' fromNode='ControlScript' toField='translation' toNode='GreenTransform'/>
<ROUTE fromField='positionTurquoise' fromNode='ControlScript' toField='translation' toNode='TurquoiseTransform'/>
<ROUTE fromField='orientationRed' fromNode='ControlScript' toField='rotation' toNode='RedTransform'/>
<ROUTE fromField='orientationGreen' fromNode='ControlScript' toField='rotation' toNode='GreenTransform'/>
<ROUTE fromField='orientationTurquoise' fromNode='ControlScript' toField='rotation' toNode='TurquoiseTransform'/>
<!-- Regular piston engine -->
See Script details on next slide...
TouchSensor isActive
129:16
INS

Edit Script

containerField

children

DEF ControlScript

USE

directOutput mustEvaluate

url array

field definitions

name	type	accessType	value
angle	SFFloat	inputOnly	
positionRed	SFVec3f	outputOnly	
positionGreen	SFVec3f	outputOnly	
positionTurquoise	SFVec3f	outputOnly	
orientationRed	SFRotation	outputOnly	
orientationGreen	SFRotation	outputOnly	
orientationTurquoise	SFRotation	outputOnly	

ECMAscript source code

```
<![CDATA[
ecmascript:
//  The function angle() receives the currently interpolated rotation angle,
//  and then computes the position and orientation of each of the cones.

function angle (value)
{
  positionRed       = new SFVec3f (Math.cos (value), 1.5 * Math.sin(value), .5);
  positionGreen     = new SFVec3f (Math.cos (value+2.094), 1.5 * Math.sin(value+2.094),   0);
  positionTurquoise = new SFVec3f (Math.cos (value+4.189), 1.5 * Math.sin(value+4.189), -.5);

  // The values 2.094 and 4.189 are 1/3 and 2/3 of 2*pi radians.
]]>
```

OK Cancel Help

Example: ScriptComplexStateEvents.x3d

Pushbutton switch changes 3-way lamp color

- TouchSensor detects button select, acts as trigger
- TimeSensor animates pushbutton cylinder
- Script invoked, checks current state to decide logic
  - Adds one to count each time, modifies color
  - Resets and turns off lamp once maximum reached

Script code kept in external file

- Still has access to all defined fields from within code
- Don't include `ecmascript:` header when external

X3D event model still governs animation flow

ScriptComplexStateEvents.x3d - Editor

ScriptComplexStateEvents.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
  <head>
    <meta content='ScriptComplexStateEvents.x3d' name='title'/>
    <meta content='A three-way lamp controlled by a Script node.' name='description'/>
    <meta content='Leonard Daly and Don Brutzman' name='creator'/>
    <meta content='10 June 2006' name='created'/>
    <meta content='2 March 2008' name='modified'/>
    <meta content='http://X3dGraphics.com' name='reference'/>
    <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
    <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptComplexStateEvents.x3d' name='identifi
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Background skyColor='1 1 1'/>
    <Viewpoint description='Control 3-way lamp' orientation='-0.993 0.101 -0.063 1.06' position='0.06 3.63 2.29'/>
    <Transform DEF='PushBox' translation='-2 0 0'>
      <Transform>
      <Transform>
    </Transform>
    <Transform DEF='Lamp' translation='1 0 0'>
      <Transform>
      <Transform DEF='Bulb' translation='0 .5 0'>
        <Shape>
          <Appearance>
            <Material DEF='LightBulbMaterial' diffuseColor='.4 .4 .4'/>
          </Appearance>
          <Sphere radius='.5'/>
        </Shape>
      </Transform>
    </Transform>
    <Script DEF='ControlScript' url='"ScriptComplexStateEvents.js" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScrip
      <field accessType='inputOnly' name='buttonMotionDone' type='SFBool'/>
      <field accessType='initializeOnly' name='buttonPushCount' type='SFInt32' value='0'/>
      <field accessType='outputOnly' name='lightColor' type='SFColor'/>
    </Script>
    <ROUTE fromField='isActive' fromNode='ButtonTimer' toField='buttonMotionDone' toNode='ControlScript'/>
    <ROUTE fromField='lightColor' fromNode='ControlScript' toField='diffuseColor' toNode='LightBulbMaterial'/>
  </Scene>
</X3D>
```

67:14 INS

ScriptComplexStateEvents.js - Editor

ScriptComplexStateEvents.js

```
// Filename:    ScriptComplexStateEvents.js
// Author:      Leonard Daly and Don Brutzman
// Identifier:  http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptComplexStateEvents.js
// Created:     10 June 2006
// Revised:     27 February 2008
// Reference:   http://www.web3d.org/x3d/content/examples/X3dSceneAuthoringHints.html#Scripts
// License:     http://X3dGraphics.com/examples/X3dForWebAuthors/license.html

function buttonMotionDone (value)
{
    if (value)
    {
        motion = new MFVec3f (new SFVec3f(0, .25, 0), new SFVec3f(0, .05, 0));
    }
    else
    {
        buttonPushCount ++;
        if (buttonPushCount > 3)
        {
            buttonPushCount = 0;
        }
        motion = new MFVec3f (new SFVec3f(0, .05, 0), new SFVec3f(0, .25, 0));
        if (buttonPushCount == 0)
        {
            lightColor = new SFColor (.2, .2, .2);
        }
        else if (buttonPushCount == 1)
        {
            lightColor = new SFColor (.6, .5, .2);
        }
        else if (buttonPushCount == 2)
        {
            lightColor = new SFColor (.7, .7, .2);
        }
        else if (buttonPushCount == 3)
        {
            lightColor = new SFColor (1, 1, .6);
        }
    }
}
```

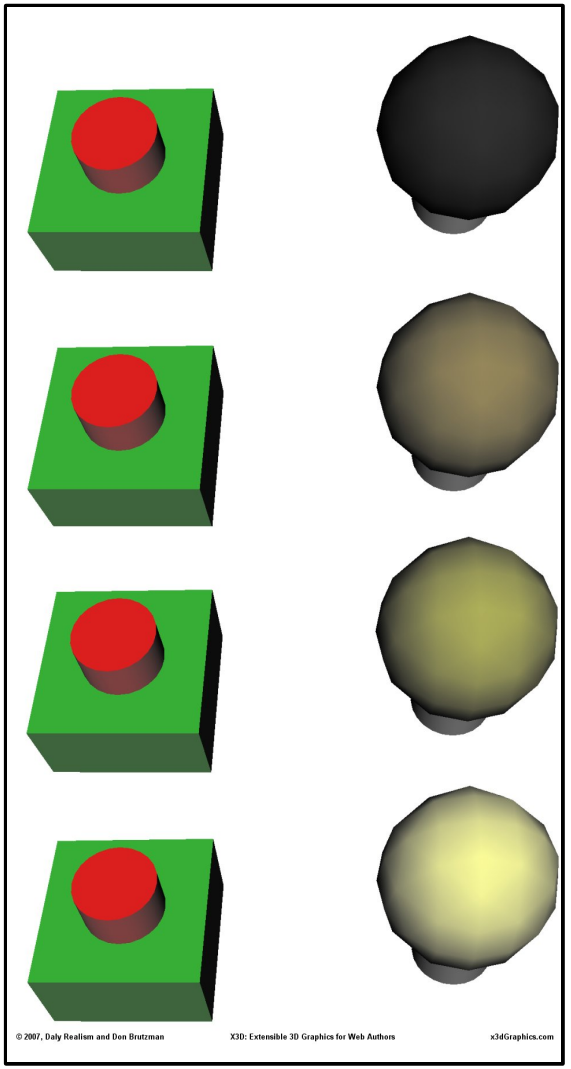
if (buttonMotionDone event value== true)
then reset next motion to go up

else (buttonMotionDone event value== false)

state variable buttonPushCount
cycles from 0 to 3, then repeats

reset next motion to go down

reset light color



1:1 INS

directOutput field

Advanced technique: SFNode, MFNode fields can give a Script node direct access to modify other nodes in the scene graph

- Usually via accessType initializeOnly

directOutput field is boolean to alert browser that such direct node referencing may produce animations without passing ROUTE events

- *directOutput*='true' indicates to browser that node references must be evaluated each time Script is invoked, since otherwise changes are likely ignored
- Otherwise *directOutput*='false' is default

mustEvaluate field

Another advanced optimization technique relates to timing of events: browsers may defer delivery of input events until output it is clear that output events are needed

- Hopefully speeding up browser performance

Resetting default value to *mustEvaluate*='true' indicates to browser that all events must be delivered immediately without delays

- Helpful for Script nodes that access the network, or else perform ongoing time-sensitive computations

initialize() and *shutdown()* methods

The *initialize()* method is invoked before scene graph rendering begins

- Good place for computing initial values, if needed
- Also helpful for *Browser.println()* tracing of setup

The *shutdown()* method is invoked after scene rendering stops, before browser exits

- Helpful location to output final results to console
- Might shutdown network connections, file reading, etc.

TestScriptInitialization.x3d - Editor

TestScriptInitialization.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specif
  <head>
    <meta content='TestScriptInitialize.x3d' name='title'/>
    <meta content='Test ability to perform Script initialize() method using ECMAsript' name='description'/>
    <meta content='Don Brutzman' name='creator'/>
    <meta content='16 February 2008' name='created'/>
    <meta content='16 August 2008' name='modified'/>
    <meta content='http://X3dGraphics.com' name='reference'/>
    <meta content='http://www.web3d.org/x3d/content/examples/X3dResources.html' name='reference'/>
    <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/TestScriptInitialize.x3d' name='identifier'/>
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Shape>
      <Text DEF='DisplayText' string='"waiting for" "Script to initialize()..."'>
        <FontStyle justify='"MIDDLE" "MIDDLE"'/>
      </Text>
      <Appearance>
        <Material ambientIntensity='0' diffuseColor='0 0 0' emissiveColor='0 0.398733 1' shininess='0.05' specularColor='0.622449 0.622449 0.622449'/>
      </Appearance>
    </Shape>
    <Script DEF='Initializer'>
      <field accessType='outputOnly' name='initializeResult' type='MFString'/>
      <![CDATA[
ecmascript:

function initialize ()
{
    scriptName = 'TestScriptInitialization.x3d';
    Browser.print ('[' + scriptName + ' initialize() commmenced...]');

    initializeResult = new MFString ("Script initialize() OK");

    Browser.print ('[...' + scriptName + ' initialize() complete]');
}
]]>
    </Script>
    <ROUTE fromField='initializeResult' fromNode='Initializer' toField='string' toNode='DisplayText'/>
  </Scene>
</X3D>
```

32:20 INS

Comparison of event and field control

ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeEventOutControl-EcmaScriptSnapshots.html

- Shows event control from *initialize()* to user interaction to *shutdown()*

ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeFieldControl-EcmaScript.x3d

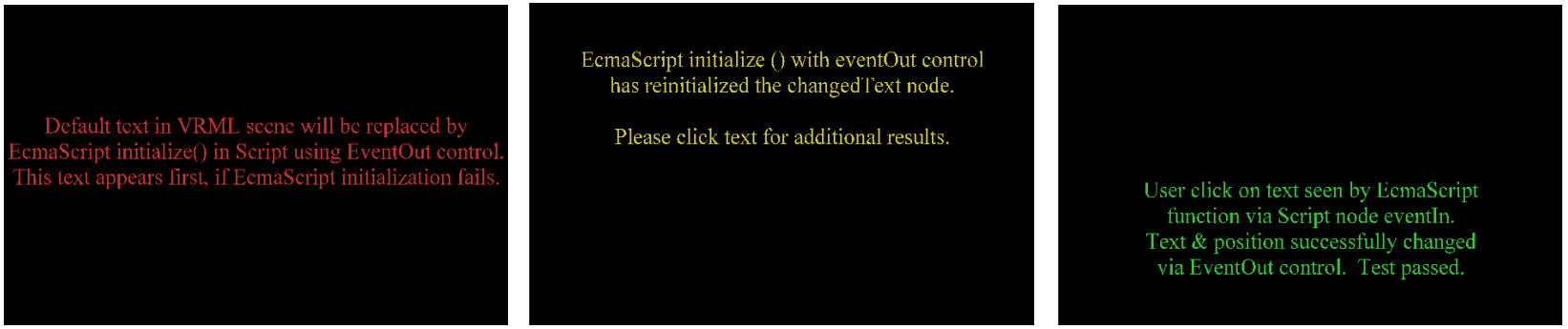
http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeFieldControl-EcmaScriptSnapshots.html

- Shows field control using passed node references, from *initialize()* to user interaction to *shutdown()*

Available via Basic Examples archive

- http://www.web3d.org/x3d/content/examples/Basic

Screen snapshots of script operation



Visual operation is the same for each example

- Initial red text is the default scene setup
- Subsequent yellow text is immediately loaded during startup as a result of initialize() method
- Third set of green text is created by Script output, in response to user clicking yellow text

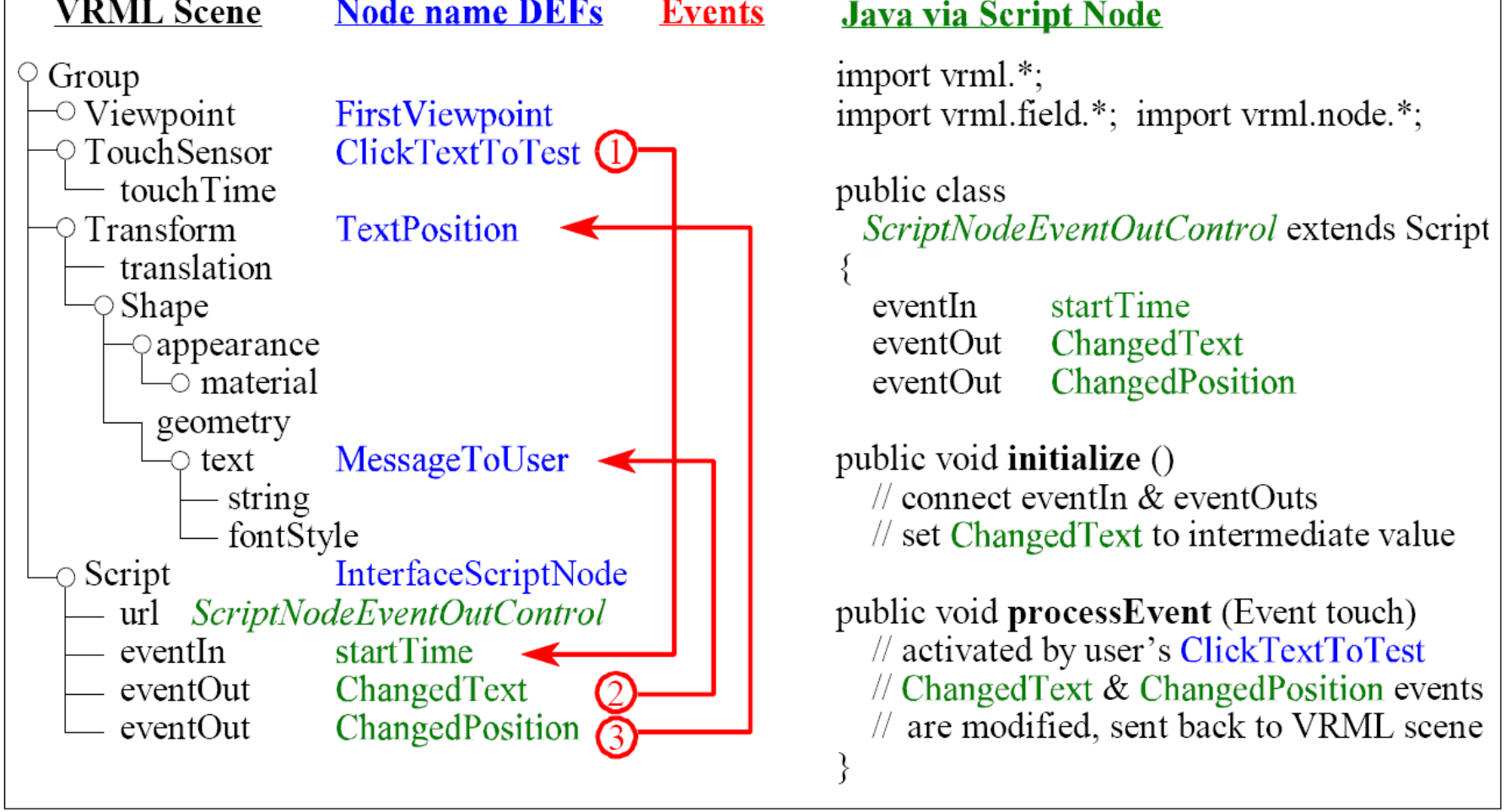


Figure 4. Script node interface between VRML and Java. This example tests event-based VRML-Java functionality. Note shared events startTime, ChangedText and ChangedPosition. The following sequence of events occurs:

(0) Initialize method on Java side establishes links, sets trace text in 3D scene to intermediate value.

(1) User clicks trace text in 3D scene with mouse, activating the TouchSensor built-in eventOut touchTime, which is ROUTEd to trigger the Script node EventIn startTime, which in turn invokes the processEvent() method in the corresponding Java class. Changed values for text and position are calculated by the Java class and then returned to the Script node as eventOut values.

(2) ChangedText eventOut sent to MessageToUser text node, sets trace text in 3D scene to final message value.

(3) ChangedPosition eventOut sent to TextPosition Transform node, moving trace text to bottom of scene.

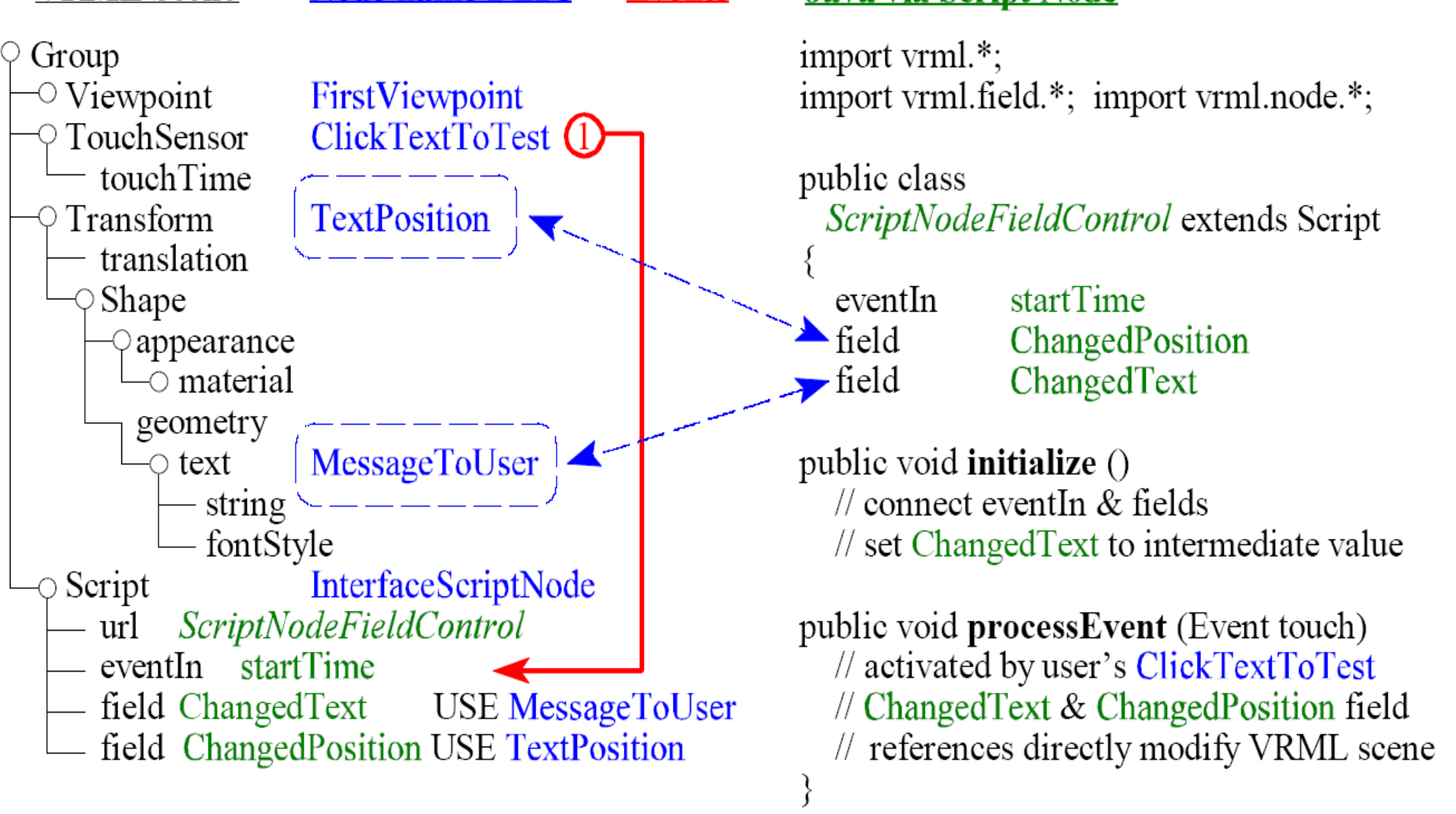


Figure 5. Field interface between VRML and Java. This example tests field-based VRML-Java functionality. Note shared event startTime, and shared fields ChangedText and ChangedPosition. Operation of this example is similar to Figure 4, except that the Java class directly manipulates VRML nodes via fields instead of sending events.

prepareEvents() and *eventsProcessed()* methods

prepareEvents() method is invoked at the beginning of each event loop, before other ROUTE processing occurs

- Helpful for advance scripting techniques

eventsProcessed() method can combine handling and response when multiple events arrive

- Useful for avoiding confusion about which event might arrive first or second, ensures all values received and ready for computation
- Only occurs once during each event loop

Browser functions 1

Function Name	Returns
Browser.getName	Name of the X3D browser.
Browser.getVersion	Provides the version identifier of the X3D browser.
Browser.getCurrentSpeed	Provides the navigation speed of the current world, obtained from currently bound NavigationInfo node, in meters/second for the coordinate system of the currently bound Viewpoint.
Browser.getCurrentFrameRate	Provides the current display update rate in frames/second.
Browser.getSupportedProfiles	Lists all profile names supported by the browser, for example "Interactive" "Immersive" "Full"
Browser.getSupportedComponents	Lists all supported component names.
Browser.createScene	Creates a new, empty scene that conforms to the provided profile and component declarations.
Browser.replaceWorld	Replaces the current world with the world provided as a parameter.
Browser.importDocument	Imports a W3C Document Object Model (DOM) document, or document fragment, and converts it to an X3D scene.
Browser.loadURL	Inserts the content identified by the url value into the current world.

Browser functions 2

Browser.setDescription	Sets the description title in the browser title bar, if available.
Browser.createX3DFromString	Creates X3D nodes from a string input.
Browser.createX3DFromStream	Creates X3D nodes from a network stream input.
Browser.createX3DFromURL	Creates X3D nodes from a file referred to by an url.
Browser.getRenderingProperties	Provides a list of the current hardware-rendering capabilities available to the browser.
Browser.getBrowserProperties	Provides a list of the current functional capabilities provided by the browser.
Browser.changeViewpoint	Changes the currently bound viewpoint based on input value Next, Previous, First, or Last.
Browser.print	Prints a string message to the browser's console.
Browser.dispose	Indicates that the client is about to exit, and the browser can dispose any consumed resources.

Script	Script provides programmed behavior for a scene. Define the script interface with <field> tags. Scripting code is embedded in a child CDATA node or (deprecated) in the url field. Optionally supported languages: ECMAScript/JavaScript and (via url to a myNode.class file) Java.
DEF	**[DEF ID #IMPLIED]** DEF defines a unique ID name for this node, referencable by other nodes. **Hint:** DEF name is needed or else ROUTEs cannot connect Script node interfaces. **Hint:** descriptive DEF names improve clarity and help document a model.
USE	**[USE IDREF #IMPLIED]** USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. **Hint:** USEing other geometry (instead of duplicating nodes) can improve performance. **Warning:** do NOT include DEF (or any other attribute values) when using a USE attribute!
url	**[url: accessType inputOutput, type MFString CDATA #IMPLIED]** points to a script file or contains scripting code preferred alternative to url scripts: insert a CDATA node to contain embedded source code CDATA can protect literals like < and > from syntax checkers. **Hint:** ECMAScript is the same as JavaScript.
directOutput	**[directOutput: accessType initializeOnly, type SFBool (true|false) "false"]** Set directOutput true if Script has field reference(s) of type SFNode/MFNode, and also uses direct access to modify attributes of a referenced node in the Scene. **Hint:** set directOutput true if Script dynamically establishes or breaks ROUTEs. **Hint:** directOutput is a browser hint to avoid overoptimizing referenced nodes, since the Script may change their attribute values without ROUTEd events. **Hint:** directOutput false means Script cannot modify referenced nodes or change ROUTEs.
mustEvaluate	**[mustEvaluate: accessType initializeOnly, type SFBool (true|false) "false"]** If mustEvaluate false, then browser may delay sending input events to Script until outputs are needed. If mustEvaluate true, then Script must receive input events immediately without browser delays. **Hint:** set mustEvaluate true when sending/receiving values via the network.
containerField	**[containerField: NMTOKEN "children"]** containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	**[class CDATA #IMPLIED]** class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

field	**A field element defines an interface attribute or node.** **Hint: first add Script, ProtoDeclare or ExternProtoDeclare before adding a field.** **Hint: put initializing SFNode/MFNode into contained content.**
name	**[name: NMTOKEN #REQUIRED]** Name of this field variable.
accessType	**[accessType: (inputOnly|outputOnly|initializeOnly|inputOutput) #REQUIRED]** Event-model semantics for field set/get capabilities. Hint for VRML 97: inputOnly=eventIn, outputOnly=eventOut, initializeOnly=field, inputOutput=exposedField. **Warning:** inputOutput=exposedField not allowed in VRML 97 Script nodes, use initializeOnly=field for backwards compatibility.
type	**[type: (select from types list) #REQUIRED]** Base type of this field variable.
value	**[value: outputOnly CDATA #IMPLIED]** Provide default initialization value for this field variable (may be later re-initialized by ProtoInstance fieldValue). **Hint:** SFNode/MFNode are initialized using contained content, instead of this value attribute. **Hint:** required for Script and ProtoDeclare. **Warning:** not allowed for ExternProtoDeclare. **Warning:** not allowed by inputOnly or outputOnly variables.
appinfo	**[appinfo type SFString CDATA #IMPLIED]** Application information to provide simple description usable as a tooltip, similar to XML Schema appinfo tag.
documentation	**[documentation type SFString CDATA #IMPLIED]** Documentation url for further information, similar to XML Schema documentation tag.

Additional Resources

Additional Resources

AjaX3D is a proposal for use of Asynchronous Javascript for X3D

- Experimental, not part of X3D specification
- http://www.ajax3d.org

Greg Seidman, Hotpot paper and code

- http://zing.ncsl.nist.gov/~gseidman/vrml/repos
- VRML: Past, Present, and Future
  http://zing.ncsl.nist.gov/~gseidman/class/vrml.html

Chapter Summary

Summary: Event Utilities and Scripting

Event utility nodes simplify data-type conversion, creation of some ROUTE connections is easier

Sequencer nodes similar to Interpolator functions

- Interpolators: continuous (floating point) outputs
- Sequencers: discrete (boolean, integer) outputs

Numerous event-utility nodes for good flexibility

- BooleanFilter, BooleanSequencer, BooleanToggle, BooleanTrigger
- IntegerSequencer, IntegerTrigger, TimeTrigger

Script node encapsulates ECMAScript, Java code

Suggested exercises

Draw, build event chains with Event Utility nodes

- Combination of boolean and time converters, especially in combination with TimeSensor
- IntegerSequencer, IntegerTrigger with Switch node
- Hey, how about a "Rube Goldberg" animation?! ☺

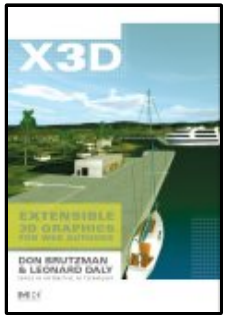
Build Script node to convert between data types

- Example: StringSensor input for rotation angle, converted to SFRotation about y-axis
- Utilize `initialize()`, `Browser.print()` and `Browser.println()` functions to provide trace statements for debugging

References

References 1

X3D: Extensible 3D Graphics for Web Authors
by Don Brutzman and Leonard Daly, Morgan Kaufmann Publishers, April 2007, 468 pages.



- Chapter 9, Event Utilities and Scripting
- http://x3dGraphics.com
- http://x3dgraphics.com/examples/X3dForWebAuthors

X3D Resources

- http://www.web3d.org/x3d/content/examples/X3dResources.html

References 2

X3D-Edit Authoring Tool

- https://savage.nps.edu/X3D-Edit

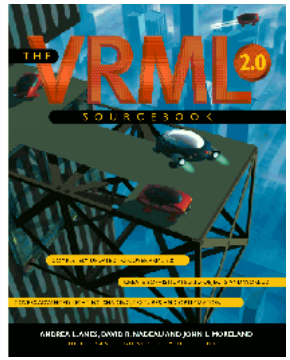
X3D Scene Authoring Hints

- http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html

X3D Graphics Specification

- http://www.web3d.org/x3d/specifications
- Also available as help pages within X3D-Edit

References 3



VRML 2.0 Sourcebook by Andrea L. Ames, David R. Nadeau, and John L. Moreland, John Wiley & Sons, 1996.

- http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm
- http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook
- Chapter 30 - Scripts

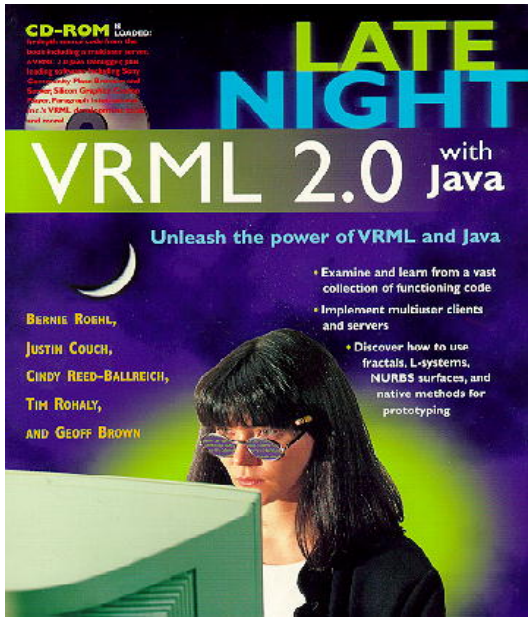
ECMAScript ECMA-262 Specification

- http://www.ecma-international.org/publications/standards/Ecma-262.htm
- http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf
- http://www.web3d.org/specifications/Ecma-262.pdf

References 4

Late Night VRML 2.0 with Java

- by Bernie Roehl, Justin Couch, Cindy Reed-Ballreich, Tim Rohaly and Geoff Brown
- Ziff-Davis Press (Macmillan Publishers), 1997
- http://www.ece.uwaterloo.ca/~broehl/vrml/lnvj



Contact

Don Brutzman

brutzman@nps.edu

http://faculty.nps.edu/brutzman

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA
1.831.656.2149 voice

web|3D CONSORTIUM

CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- http://cgems.inesc.pt

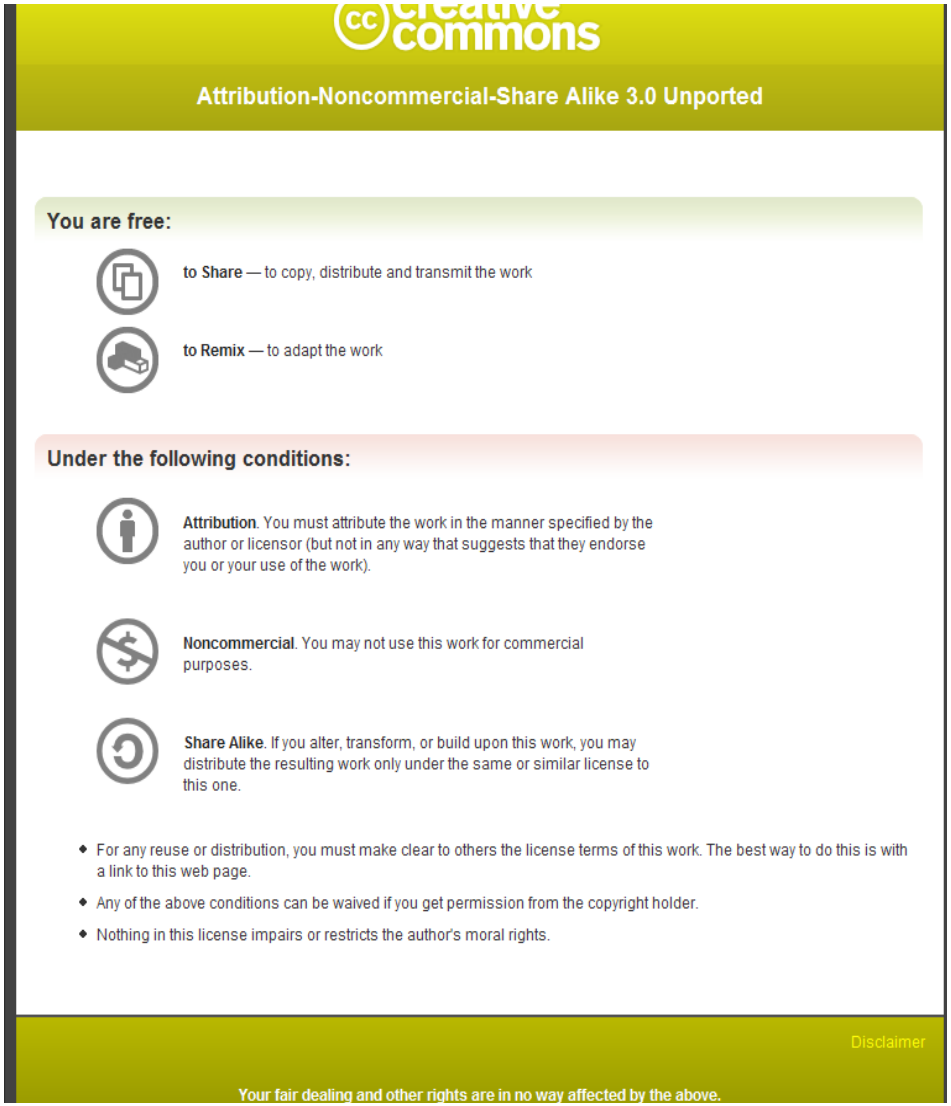
X3D for Web Authors recognized by CGEMS! ☺

- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics

Creative Commons open-source license

http://creativecommons.org/licenses/by-nc-sa/3.0



Open-source license
for X3D-Edit software and X3D example scenes

http://www.web3d.org/x3d/content/examples/license.html

X3D Graphics for Web Authors

Chapter 9

Event Utilities and Scripting

Action is eloquence.

William Shakespeare, *Coriolanus*, Act III Scene II

web3D CONSORTIUM

1

http://shakespeare.mit.edu/coriolanus/coriolanus.3.2.html

MENENIUS
Noble lady!
Come, go with us; speak fair: you may salve so,
Not what is dangerous present, but the loss
Of what is past.
VOLUMNIA
I prithee now, my son,
Go to them, with this bonnet in thy hand;
And thus far having stretch'd it--here be with them--
Thy knee bussing the stones--for in such business
Action is eloquence, and the eyes of the ignorant
More learned than the ears--waving thy head,
Which often, thus, correcting thy stout heart,
Now humble as the ripest mulberry
That will not hold the handling: or say to them,
Thou art their soldier, and being bred in broils
Hast not the soft way which, thou dost confess,
Were fit for thee to use as they to claim,
In asking their good loves, but thou wilt frame
Thyself, forsooth, hereafter theirs, so far
As thou hast power and person.

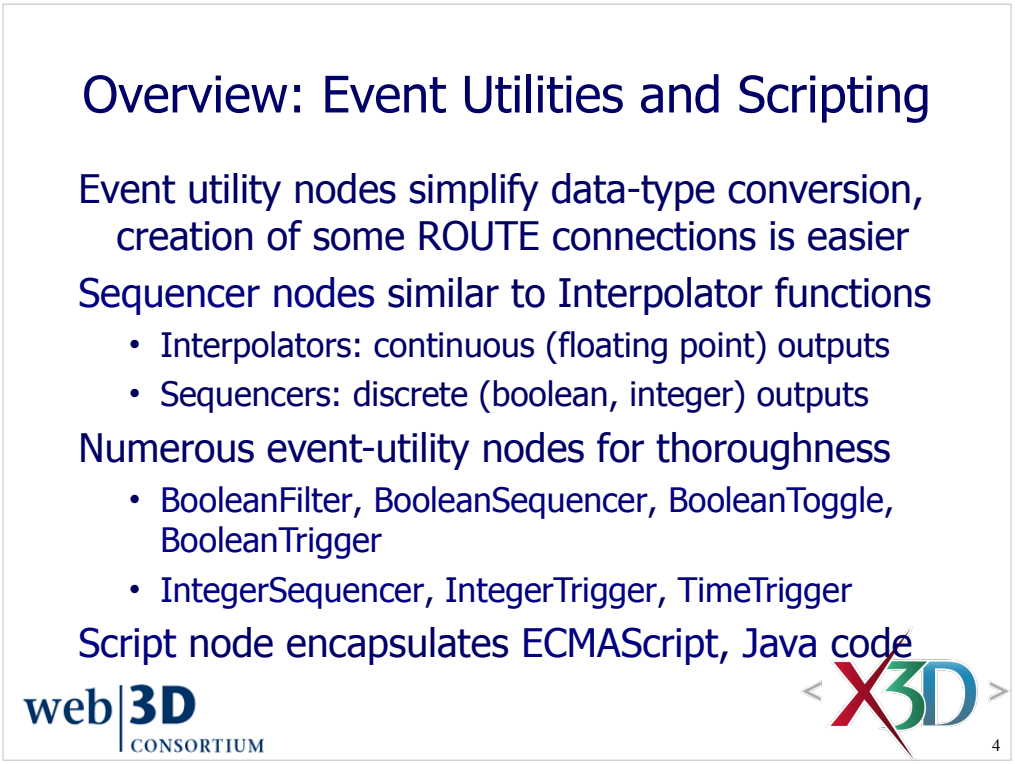
Contents

web|3D CONSORTIUM

2

Chapter Overview

3



Interpolator nodes are described in Chapter 7, Event Animation and Interpolation. It is especially important to understand the 10-step process for designing and building an animation chain.

Sensor nodes (described in Chapter 8) are also used to produce events that we can ROUTE to other nodes in the scene graph.

The event utilities make it possible (and even easy) to create longer, more sophisticated event chains and behaviors.

back to Table of Contents

Concepts

Event utilities motivation 1

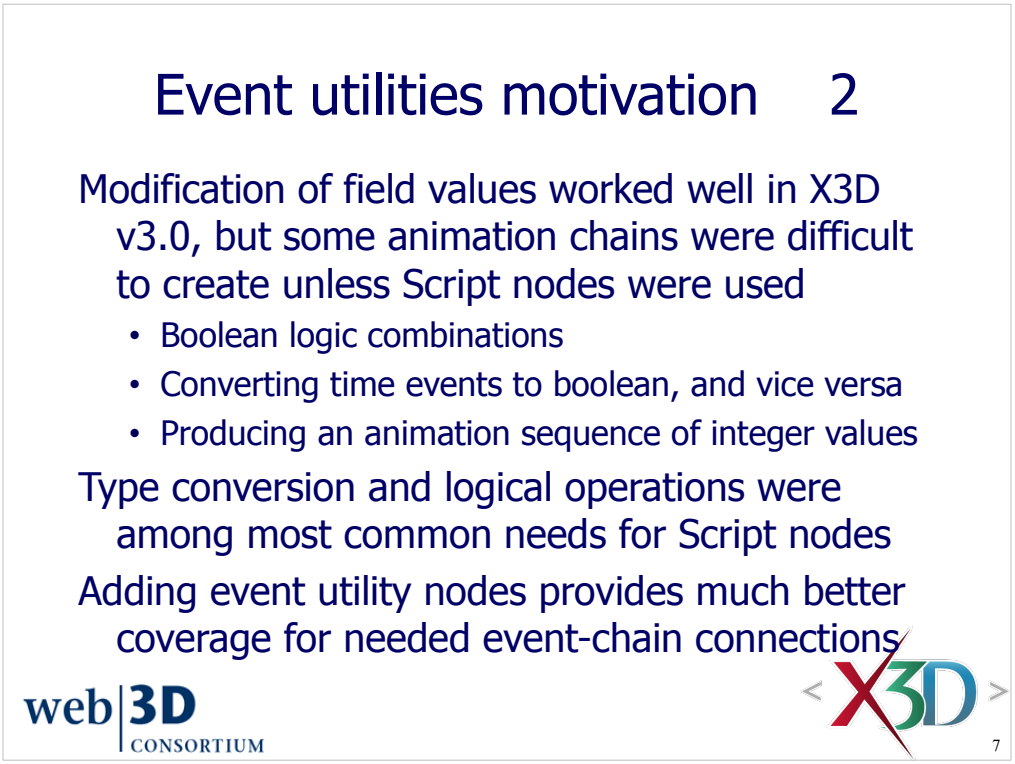
- Event ROUTE connections are fundamental to X3D animation
- Every node in the scene graph can be an animation candidate
  - Many nodes have output fields to produce values
  - Most nodes have input fields that can be changed by run-time events
  - Can connect each output-to-input pair via a ROUTE
- Certain common animation challenges led to development of event-utility node extensions to original X3D/VRML scene graph

The event utilities described in this chapter were devised to support common animation tasks. Before the sequencer, trigger and filter nodes were developed, author-written Script nodes were required for even the simplest type-conversion tasks.

Event-utilities node design and development was performed within the X3D working group. Node design work was accomplished primarily by Nick Polys, Don Brutzman and Joe Williams.

The first implementations of these nodes used Prototype definitions and Script conversions. This approach made the event-utility nodes widely testable and usable even before they were implemented in X3D browsers and approved in the specification.

Thus the development of these nodes is a good example of the extensibility of the X3D language. Building prototypes as sample implementations is an excellent way to experiment with and optimize new capabilities.



By “logical operations” we mean boolean logic: true false and or exclusive-or not etc.

Avoiding the need for Script nodes any time that a data-type conversion is needed can be considered a further motivation. Rationale:

• Script node is not included within Interactive Profile for lightweight browsers

• Script nodes take somewhat longer for new authors to learn because syntax of Script node and code is different than other native X3D nodes

• Simple is good

A good rule of thumb for authors: if you want to create animations that depend on boolean (true/false) logic or time triggers, then the Event Utility node provide the support that you need.

Advanced topic: there are still a few data types remaining in X3D that might benefit from having a dedicated utility node

• SFImage sequencer

Functional summary, comparison

- BooleanFilter and BooleanToggle simplify the negation, combination of boolean true/false logic
- BooleanSequencer and IntegerSequencer are similar to interpolator nodes, but produce single discrete values one at a time, rather than continuous stream of changing floating-point values
- BooleanTrigger, IntegerTrigger, TimeTrigger each produce a single typed-value response after receiving a triggering input
- Script node can encapsulate arbitrary functionality, using either ECMAScript (JavaScript) or Java, with resulting capabilities similar to HTML scripting

web3D CONSORTIUM

8

Declarative programming

Language is "declarative" if it describes solutions based on problem aspects, rather than directing step-by-step algorithm how to solve it

- Example: HTML for web pages
- HTML file declares *what* should appear on a page (formatted text, images, links, etc.) but does not specify *how* a browser accomplishes that rendering

Some programming languages are declarative

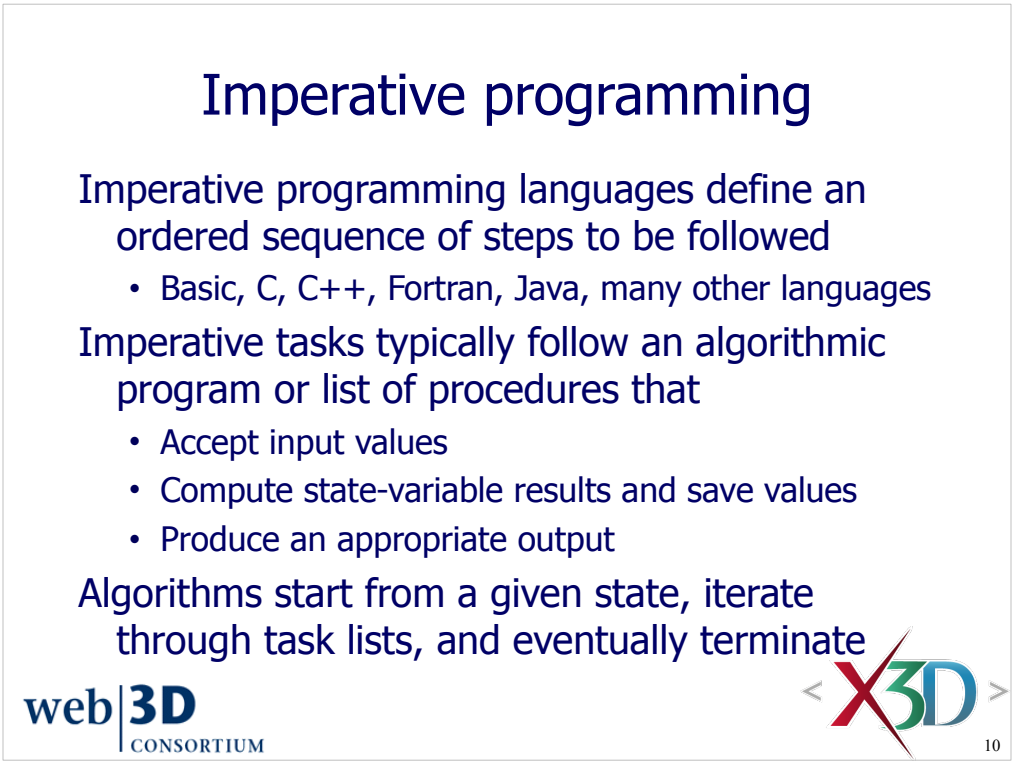
- Prolog (Programming in Logic) for logical inference
- Expert systems containing rules, relationships
- Extensible Stylesheet Language for Transformations (XSLT) processes XML inputs

Contrast declarative approaches with imperative, procedural programming.

Helpful references:

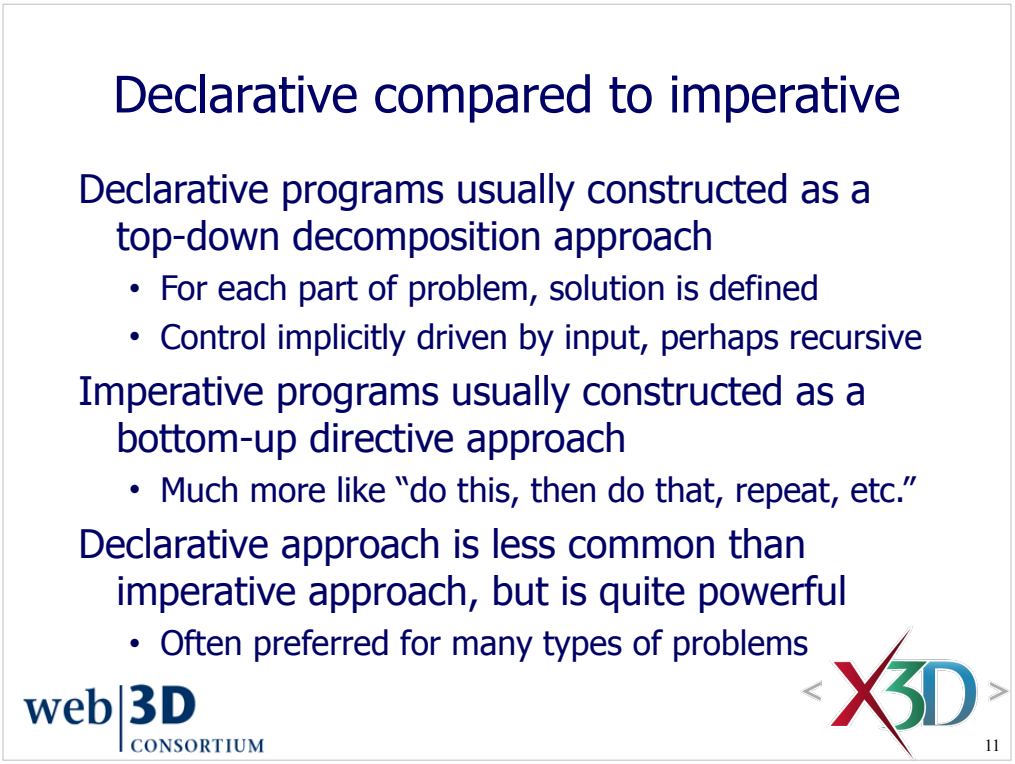
- http://en.wikipedia.org/wiki/Declarative_programming
- http://en.wikipedia.org/wiki/Imperative_programming

In general, declarative structures tend to be driven by data while procedural processes are controlled by imperative “do this, do that” command sequences.



Helpful references:

- http://en.wikipedia.org/wiki/Imperative_programming
- http://en.wikipedia.org/wiki/Algorithm

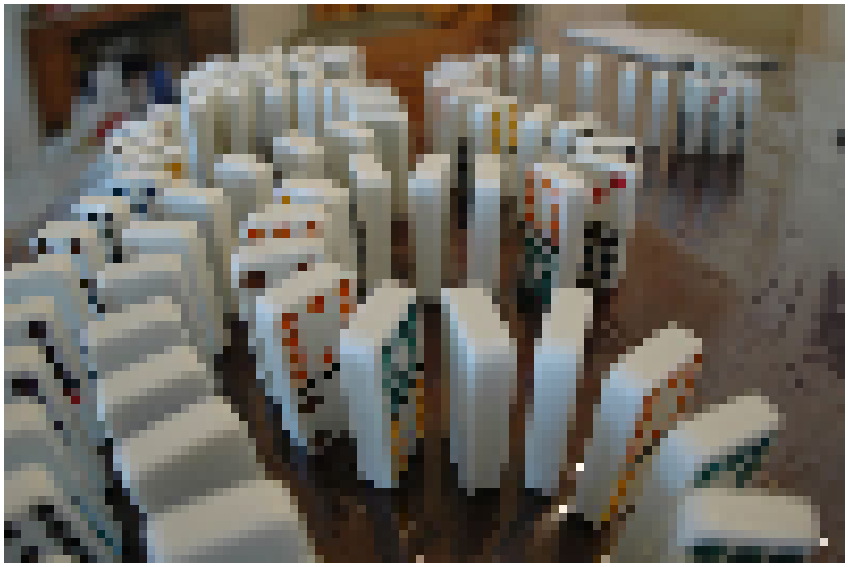


Helpful references:

- http://en.wikipedia.org/wiki/Declarative_programming
- http://en.wikipedia.org/wiki/Imperative_programming

Thought question: is HTML declarative or imperative?

Example: dominoes ready to fall



Each domino is waiting for an input stimulus, which in turn causes it to fall. There is no single controller in charge of all the dominoes, directing each in sequence. Rather the individual, aggregate behavior of the all of the dominoes combination becomes interesting when they are arranged in a fashion that lets one domino affect another.

Note that even in complex arrangements like the one shown in this image, the declarative logic for each individual domino is independent and simple:

- if sufficiently tapped, a domino tips over.

Thought question: what is the difference between the above arrangement of dominoes and another identical set of dominoes scattered on the same tabletop?

One answer is “information,” i.e. not just the locations of each domino but also the pattern of behavior that emerges when one end of the domino chain is tapped.

Really big thought question: what if the fundamental building blocks of the universe are not just matter and energy, but matter and energy and information?

Permission to use this image is gratefully acknowledged:

http://en.wikipedia.org/wiki/Image:Toppledominos.jpg

X3D scene graph is declarative

X3D design is declarative, because

- presence of various sensor and interpolator nodes,
- plus the ROUTE connections within scene graph,
- defines how scene behaves, reacting to event inputs

Can be thought of as a cause-and-effect chain

- i.e. row of dominoes ready to tip, or
- perhaps some kind of Rube Goldberg machine

Declarative thus indicates that action/reaction relationships are predeclared

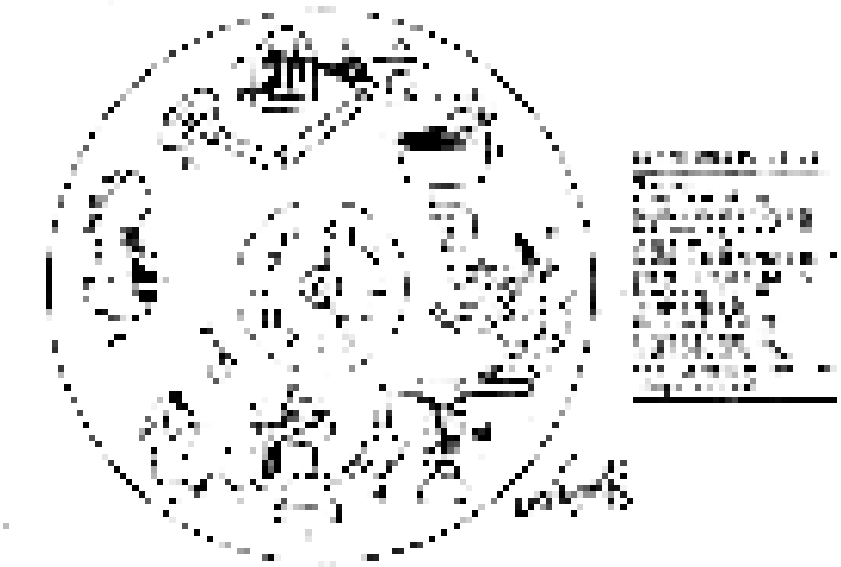
- Operates after loading and clock, interaction begins
- Input stimulus needed before output is produced

Often the necessary input stimulus is provided by one of the following trigger events:

- User interaction (through pointer selection)
- User view proximity or visibility (discussed in Chapter 12, Environment Sensors)
- Passage of computer-clock time (via TimeSensor)
- Initial loading complete (LoadSensor, or Script `initialize()` method)

Rube Goldberg: http://en.wikipedia.org/wiki/Rube_goldberg

Example: Rube Goldberg machine

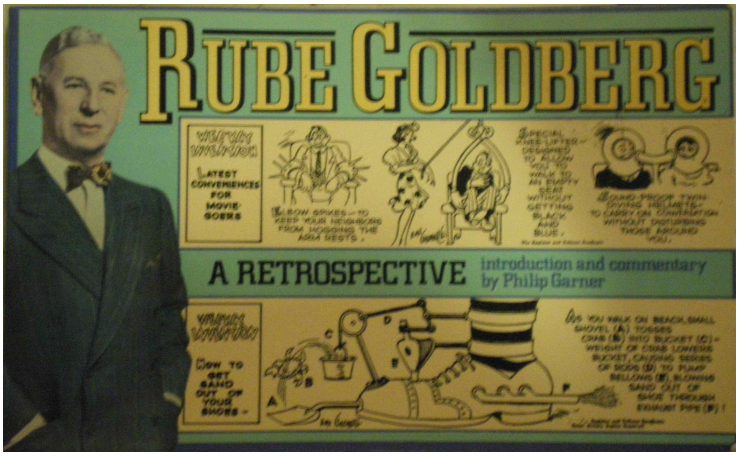


http://en.wikipedia.org/wiki/Rube_goldberg
http://www.rubegoldberg.com

Rube Goldberg Machine Contest
http://en.wikipedia.org/wiki/Rube_Goldberg_Machine_Contest
http://www.anl.gov/Careers/Education/rube/rubeinfo.html

Permission to reproduce this image requested from King Features Syndicate.
http://www.kingfeatures.com Duplication here should be OK under fair use.

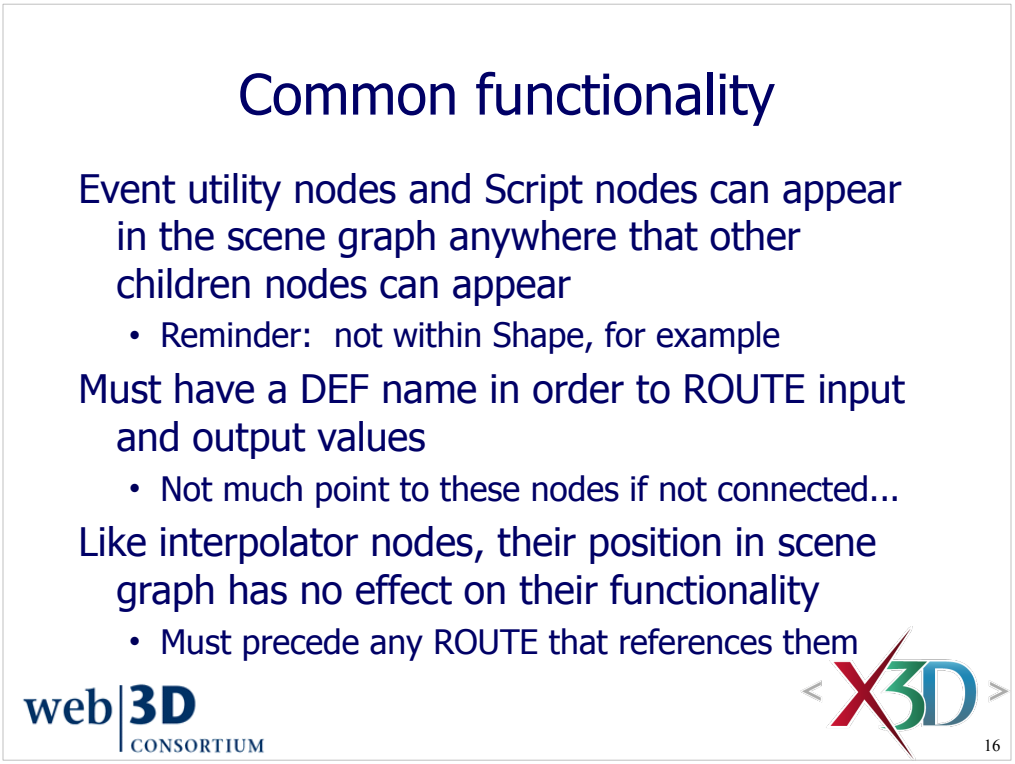
Rube Goldberg: A Retrospective. Introduction and commentary by Philip Garner. Delilah Books, distributed by Putnam Publishing Group, New York 1983.



Script node motivation: imperative

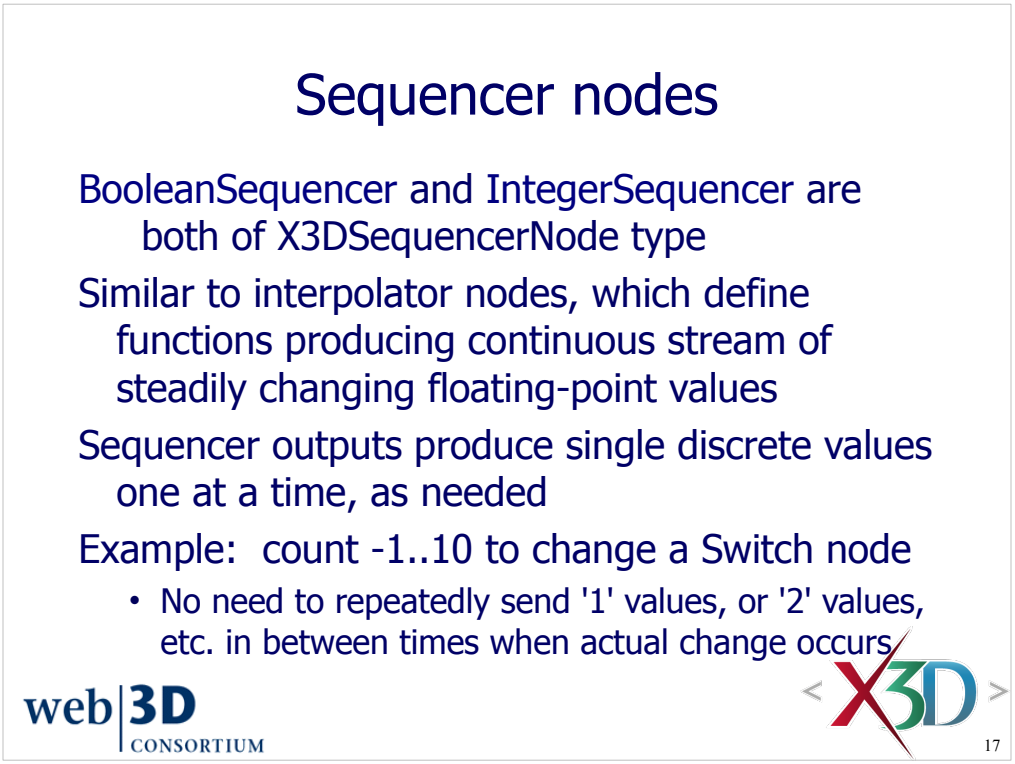
- Script node encapsulates *imperative* programs, wrapping them in an event-driven interface suitable for connection to the *declarative* approach of the X3D scene graph
  - Thus an impedance match between imperative and declarative programming paradigms
- Embedding (often small) chunks of imperative script code within a declarative event-passing X3D animation chain is powerful
  - Algorithms of arbitrary complexity can be built
  - Well-defined rules align these 2 different paradigms
  - Similar motivations for HTML use of JavaScript

Depending on the capabilities of the embedded scripting language, Script nodes can also expose special functionality like network communications, database access, scientific libraries, web services, etc.



Event utility and Script nodes are usually positioned amidst the other nodes in the animation chain using them. Proximity of these nodes and their corresponding ROUTE connections makes author debugging easier.

Some tools place all ROUTE connections at the end of a scene to avoid ordering problems, but this usually makes scene checking and diagnosing much harder.



For example: if a '3' value is been sent to a Switch node, then the child with index 3 is already selected. Resending multiple '3' values does not accomplish anything new. Thus each value in a sequence is only sent once by a sequencer node.

“Say something once... why say it again?”

> • David Byrne, Chris Frantz and Tina Weymouth, *Talking Heads,* “Psycho Killer”, 1977

Example: boolean, integer sequencers

Following example illustrates coordinated use of BooleanSequencer, IntegerSequencer nodes

- Note common TimeSensor node (named Clock)

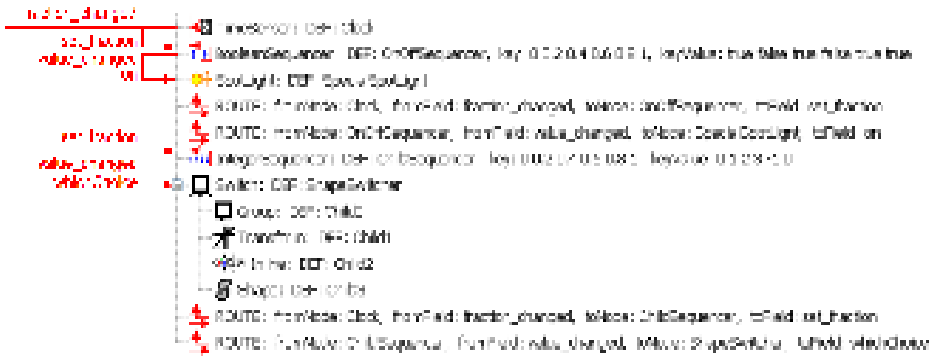


Figure 9.1, page 253, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d

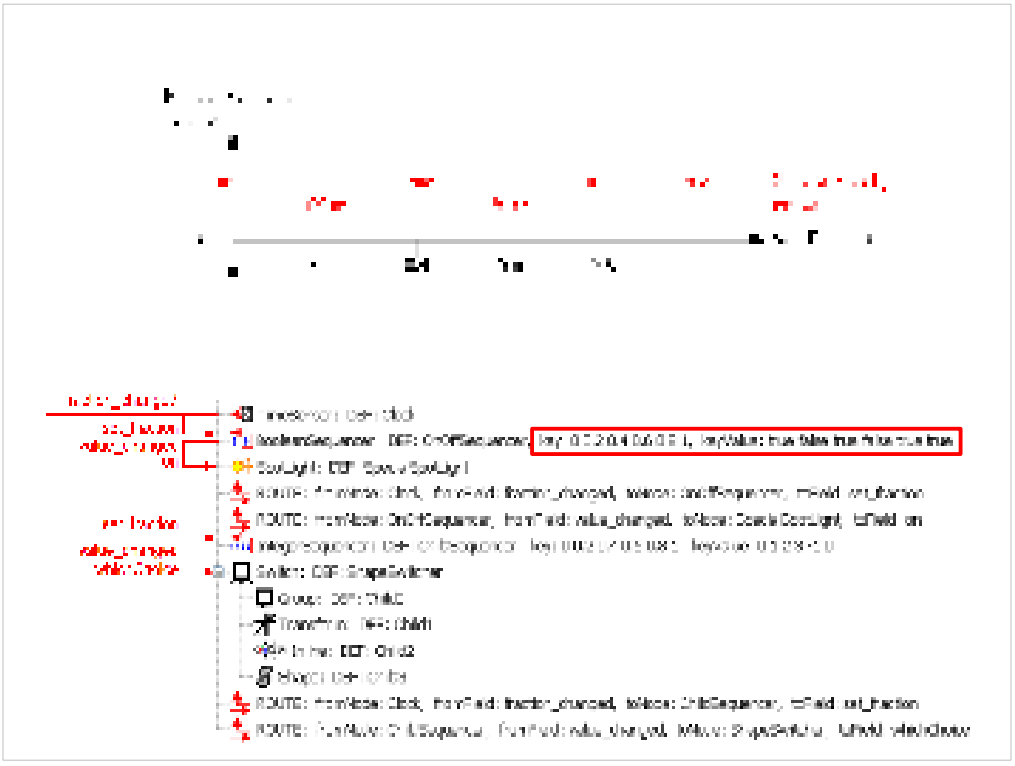


Figure 9.1, page 253, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d

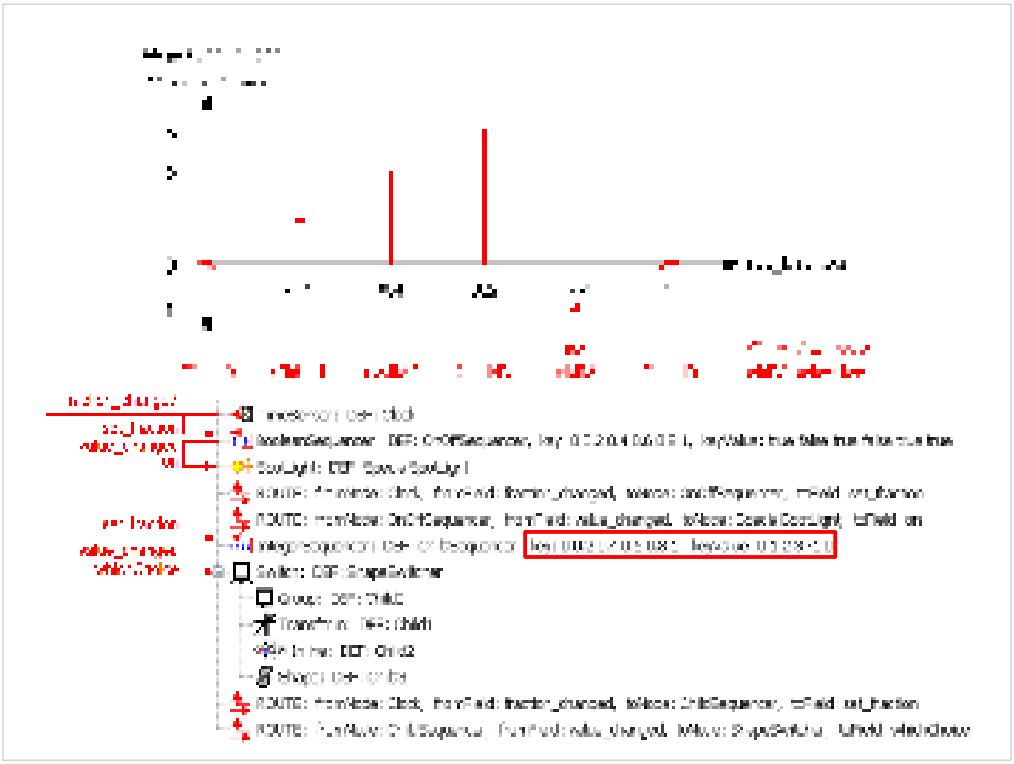


Figure 9.1, page 253, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d

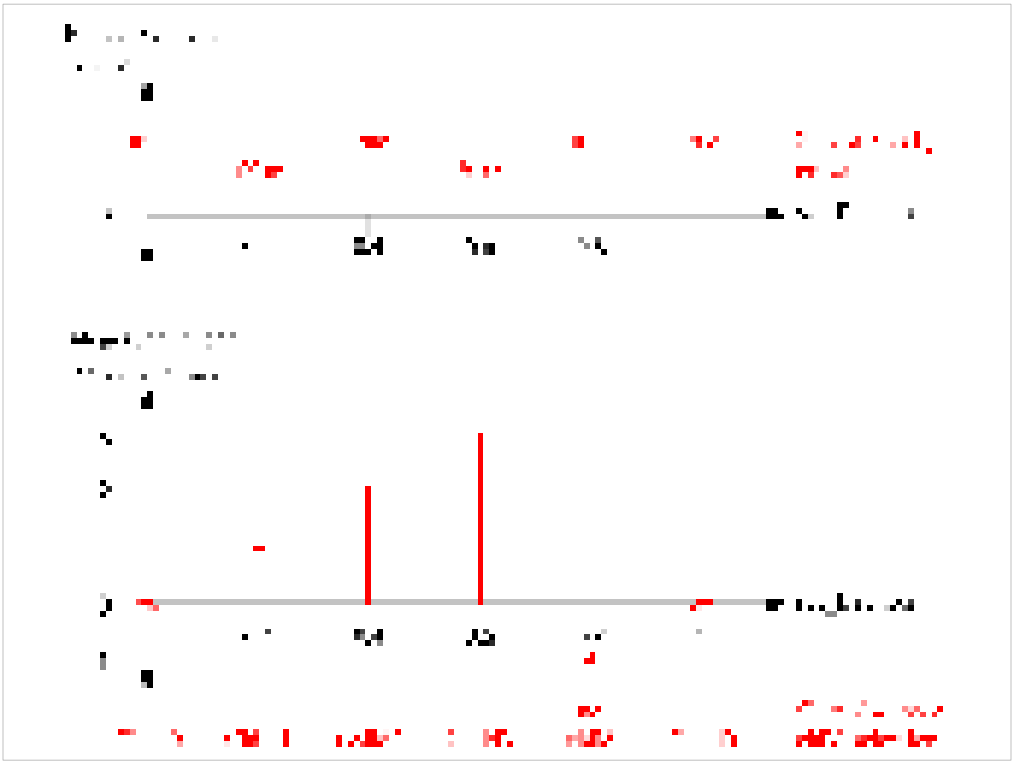
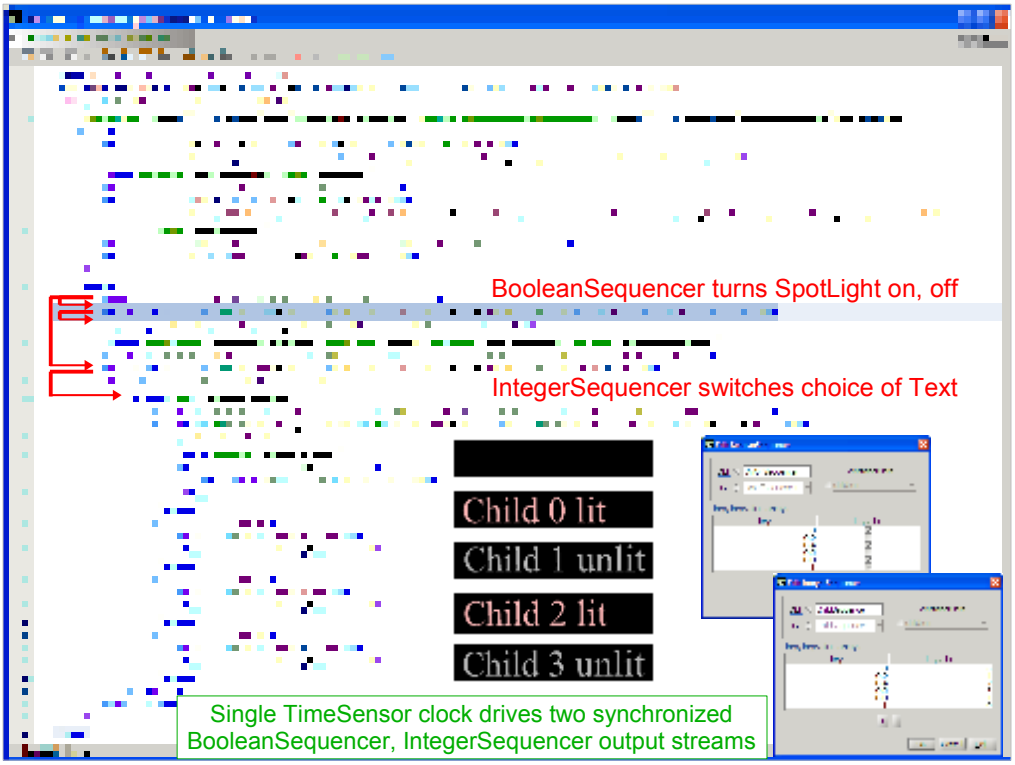


Figure 9.1, page 253, *X3D for Web Authors*

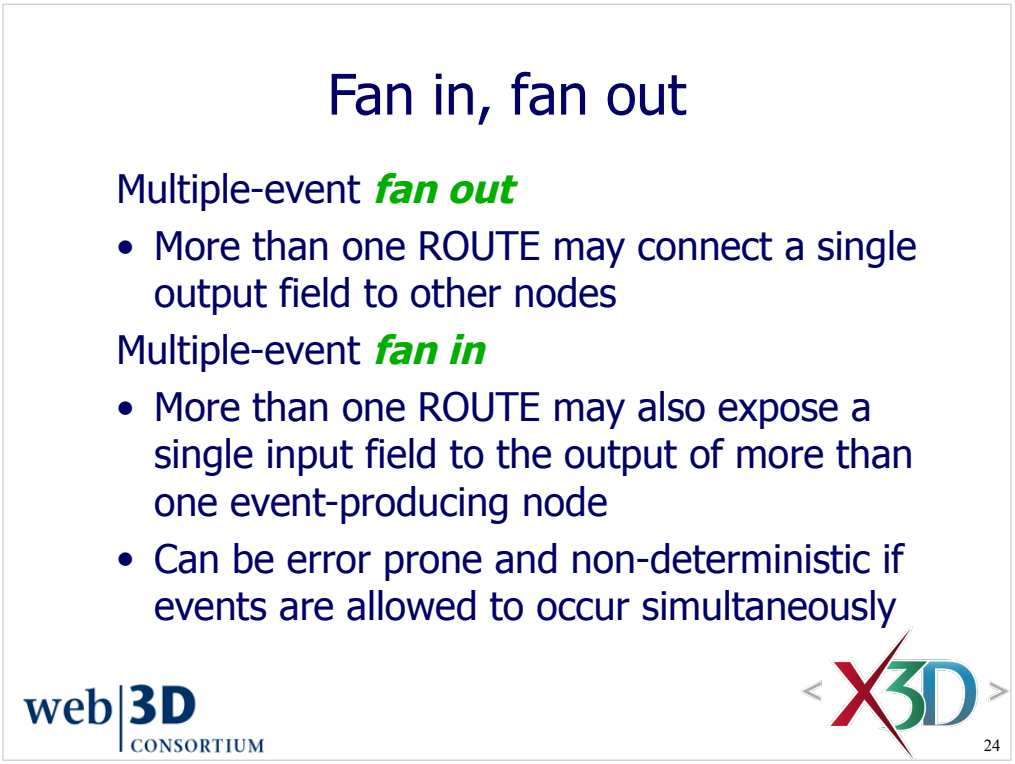
http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d

http://x3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanSequencerIntegerSequencer.x3d

The labels in the Text node for this example were changed slightly to better explain what actually happens during each step. The functionality of the scene is unchanged.



Flexible routing of multiple events to multiple nodes is often just great, leading to sophisticated (i.e. “cool”) behavior animations.

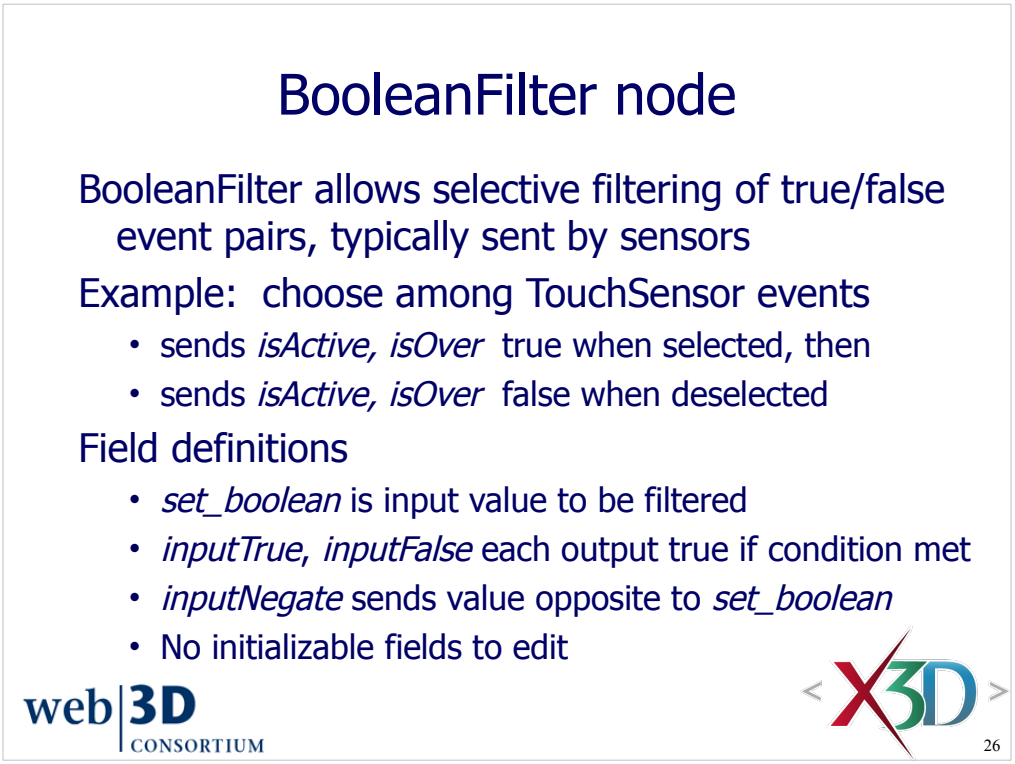
Note that multiple fan in of events simultaneously to a single target field is error prone. Animation results may be nondeterministic, because one of the input events will override another if they both occur during the same timestamp. Such overloading is usually noticed as flip-flopping or unpredictable rendering in the scene.

Helpful convenience: the X3dToXhtml pretty-print stylesheet shows ROUTE connections as comments. So you can use those to identify multiple-ROUTE fan in and fan out.

TODO: example link

X3D Nodes and Examples

25



Because there are no initializable BooleanFilter fields (with *accessType*='initializeOnly' or *accessType*='inputOutput') the editor for this node is minimal. All of the functionality for this node is exposed via careful use of ROUTE statements.

Boolean functions such as AND, OR, XOR, NAND etc. are not provided as native capabilities in X3D. However, such functions can be written using a Script node.

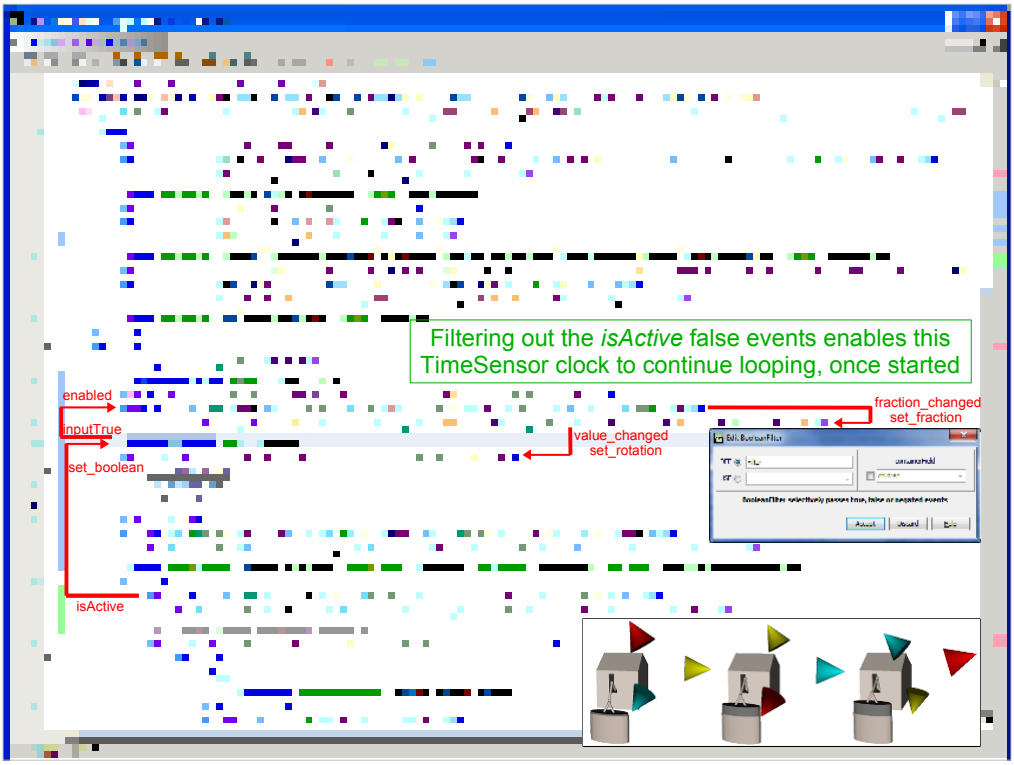
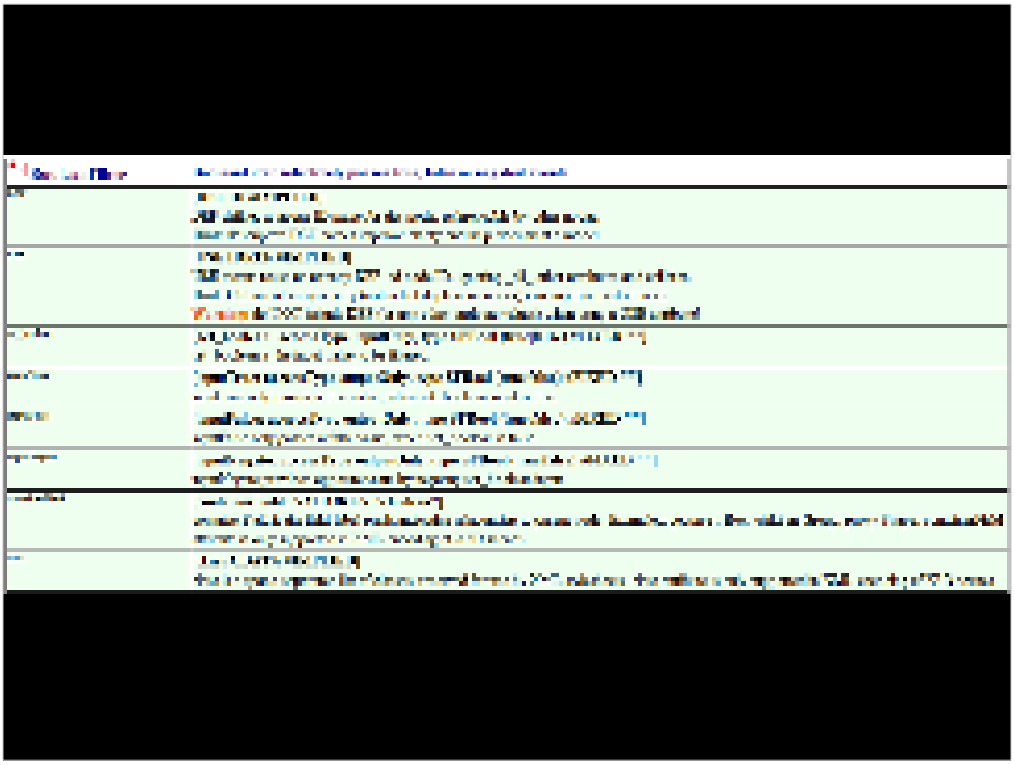


Figure 9.2, page 255, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanFilterPumpHouse.x3d

This example shows a a “turn on” switch capability with no corresponding “turn off.”



http://www.web3d.org/x3d/content/X3dTooltips.html#BooleanFilter

BooleanSequencer node

BooleanSequencer contains a *keyValue* array of boolean values that can be triggered by input *set_fraction* values for corresponding *key* array

As with interpolator nodes, output field is named *value_changed*

Useful fields: node triggering via boolean events

- SFBool inputOnly *next:* send next *keyValue*
- SFBool inputOnly *previous:* send prior *keyValue*
- Provides a helpful alternative to using the more typical TimeSensor *fraction_changed*

web3D CONSORTIUM

29

Setting up a BooleanSequencer node is similar to setting up an interpolator node. First define the functional characteristics by defining the paired *key* and *keyValue* arrays. Then expose the functionality for this node by connecting it with ROUTE statements, with input from a TimeSensor and output going to the target node of interest.

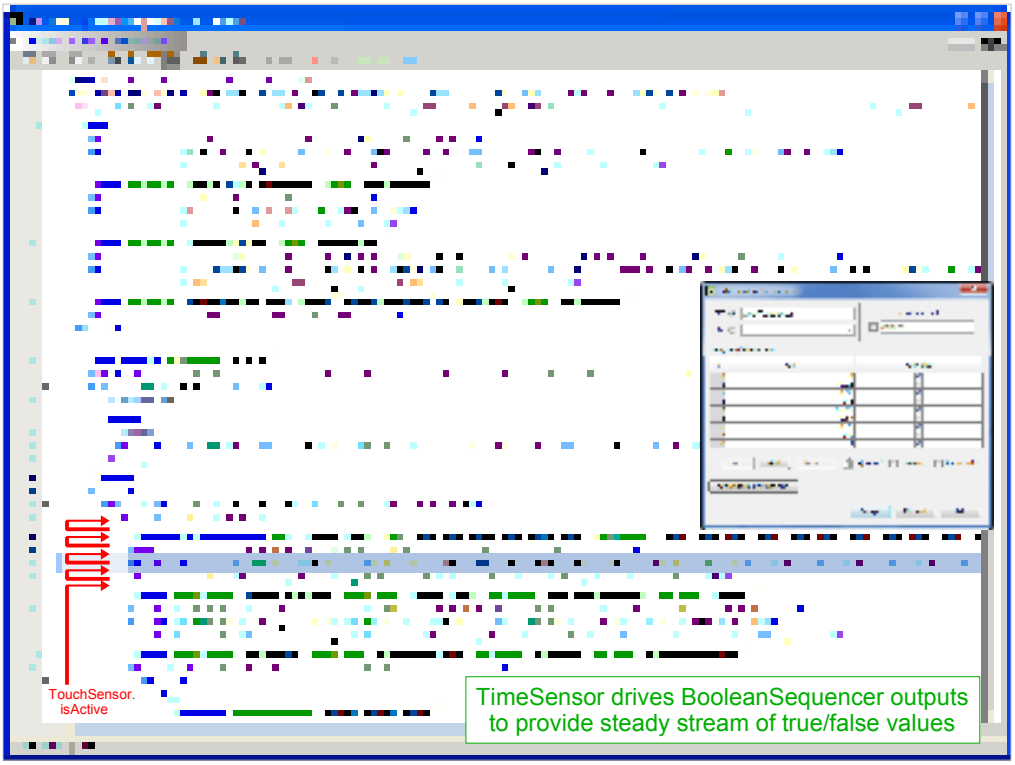
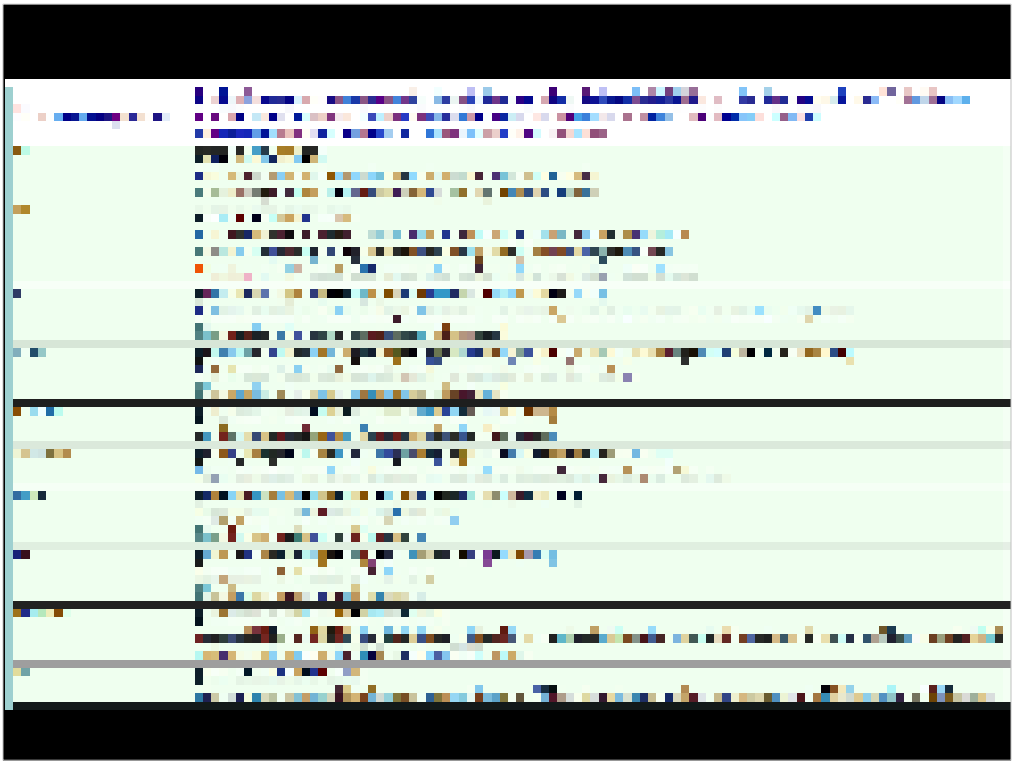


Figure 9.3, page 257, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/IntegerSequencerPumpHouse.x3d

Note that pump operation appears to stop, then jump, to a new position. This is because the TimeSensor *fraction_changed* output continues to advance (seconds since 1 January 1970) while the TimeSensor is disabled. Thus, once re-enabled, the *fraction* output value "jumps."



http://www.web3d.org/x3d/content/X3dTooltips.html#BooleanSequencer

BooleanToggle node

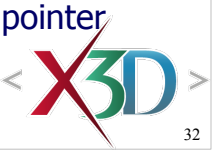
BooleanToggle remembers, can negate booleans

- Maintains inputOutput state variable named *toggle*
- Negates *toggle* value if input *set_boolean* is true
- No change occurs if input event *set_boolean* is false

Honoring input *set_boolean* true (and ignoring *set_boolean* false) is especially helpful when connected to *isActive/isOver* sensor outputs

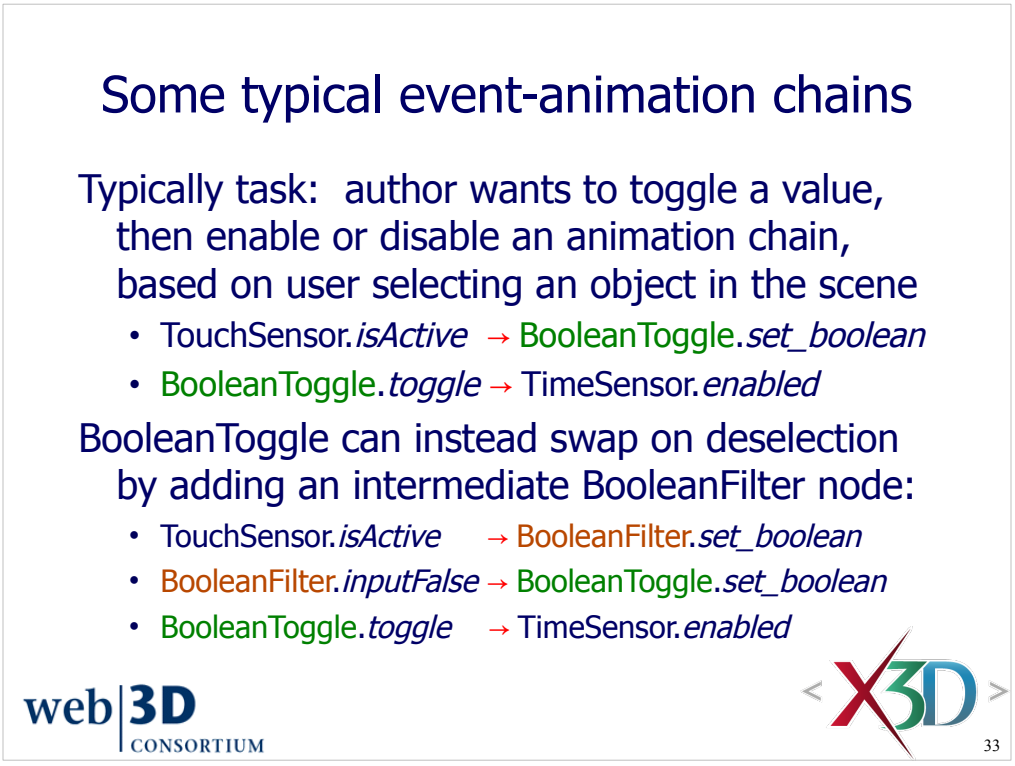
- Example: only an *isActive* true event causes change
- Toggling occurs only on selection, rather than both selection+deselection, so user can remove pointer

web3D CONSORTIUM



32

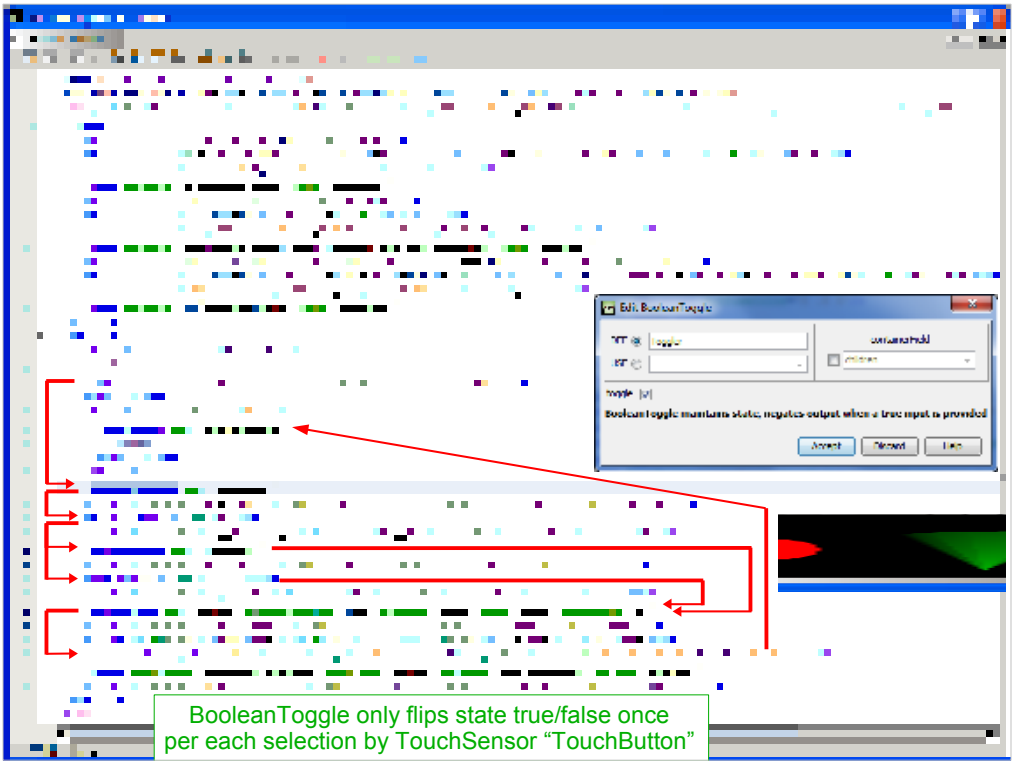
It can be frustrating for users if they click to select and change something, then have it change right back when they let go of (deselect) the geometry. In that arrangement, they can only maintain a change by keeping the pointer locked into selecting the object. So that pattern is a complete nonstarter when any other user interaction is needed.



If using a mouse:

- Selection means that user has performed initial down-click
- Deselection means that user has performed subsequent up-click

Arrow symbol → means ROUTE

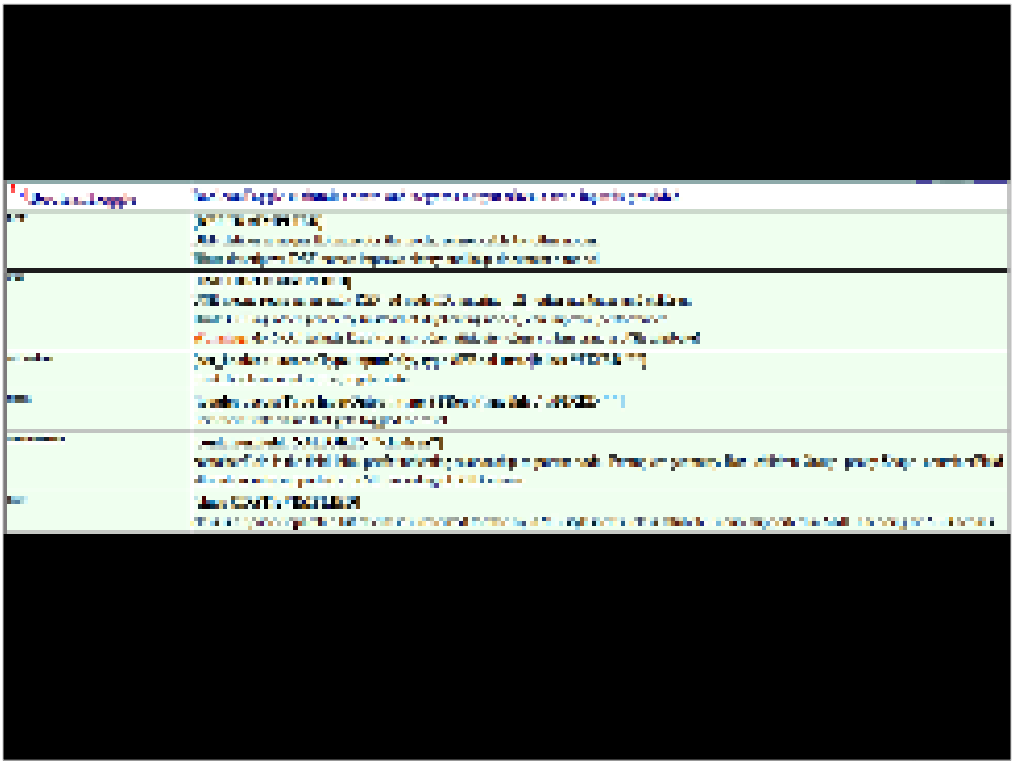


http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanToggle.x3d

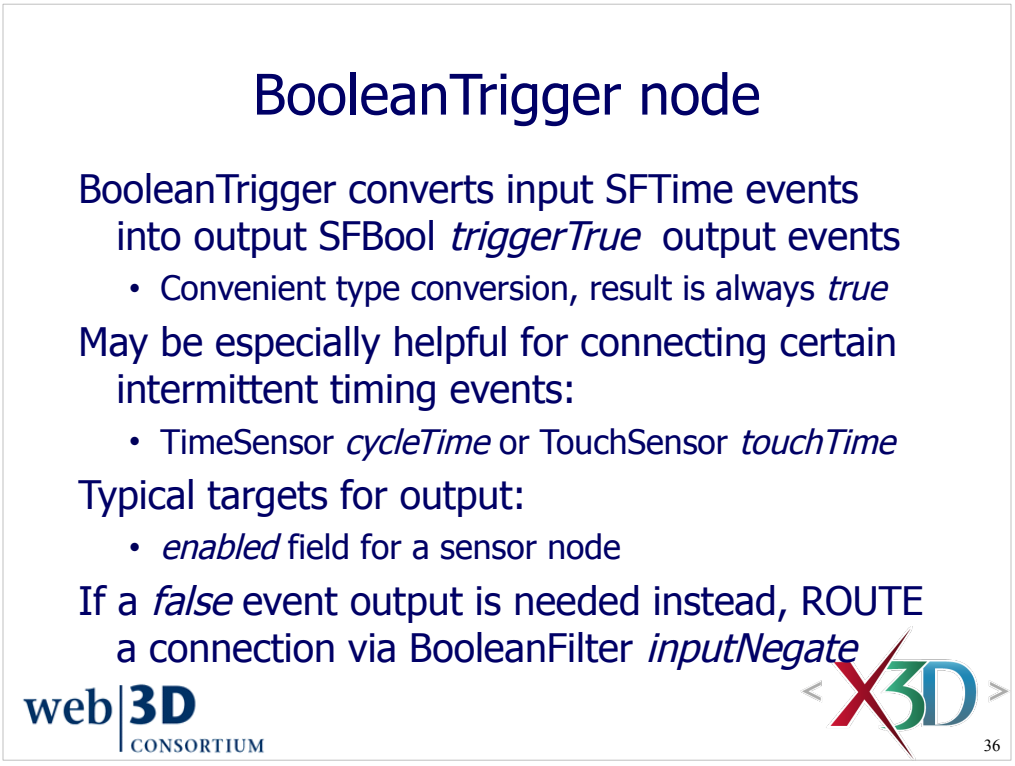
Similar example:

Figure 9.4, page 259, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanTogglePumpHouse.x3d



http://www.web3d.org/x3d/content/X3dTooltips.html#BooleanToggle



If you want a *false* event instead of a *true* event, then ROUTE the output to a BooleanFilter *set_input* field, then ROUTE the output event from the BooleanFilter *outputNegate* field. In other words, just insert the Boolean node needed into your ROUTE chain that makes your desired logic work.

Because there are no initializable BooleanTrigger fields (*accessType*='initializeOnly' or *accessType*='inputOutput') the editor for this node is minimal. All of the functionality for this node is exposed via careful connection of ROUTE statements.

From X3D specification 30.4.4 BooleanTrigger:

> BooleanTrigger is a trigger node that generates Boolean events upon receiving time events. The triggerTrue event is generated when the BooleanTrigger receives a set_triggerTime event. The value of triggerTrue shall always be TRUE.

Node comparison:

- BooleanTrigger (time → boolean)
- IntegerTrigger (boolean → integer)
- TimeTrigger (boolean → time)

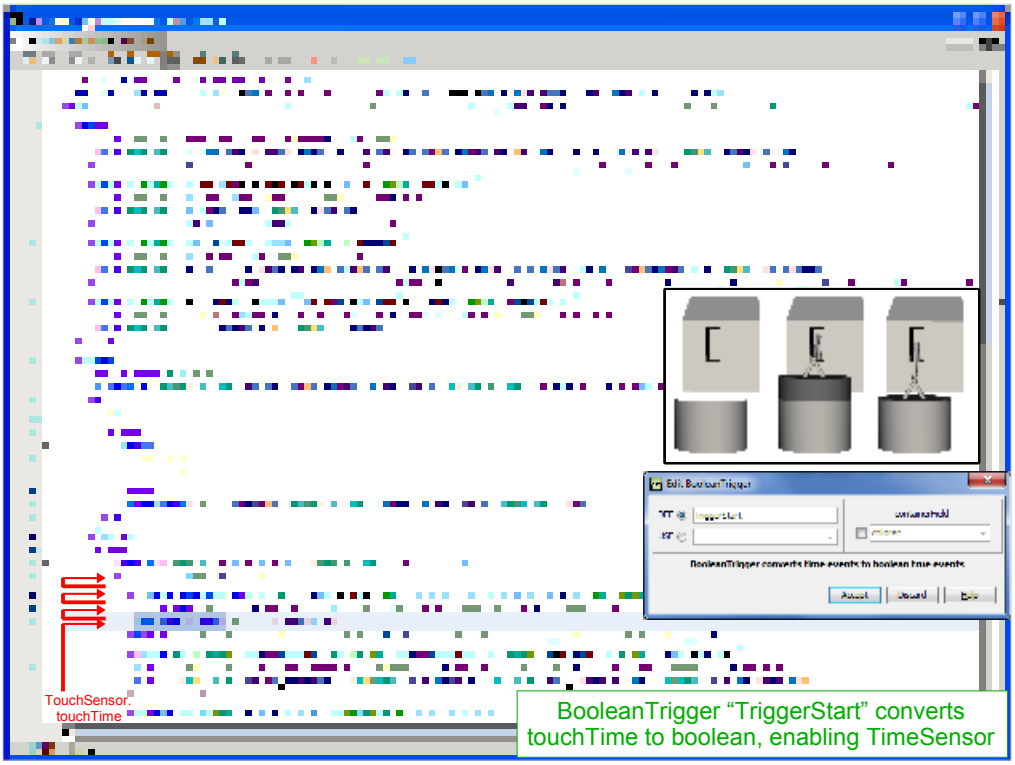
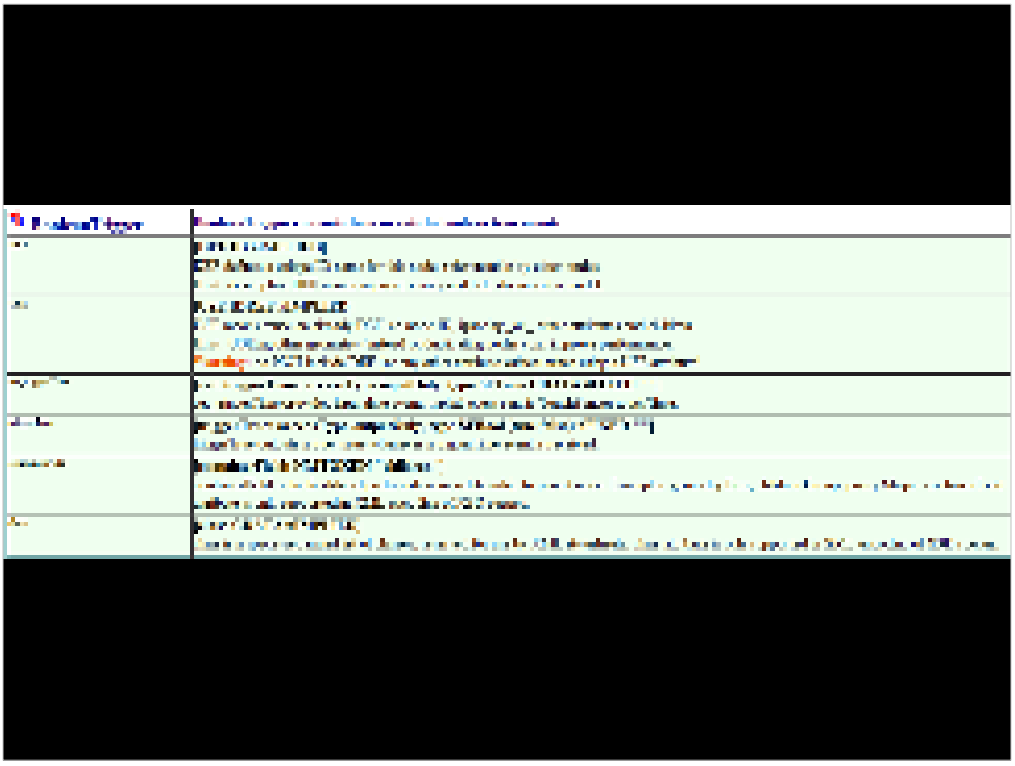


Figure 9.5, page 260, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/BooleanTriggerPumpHouse.x3d



http://www.web3d.org/x3d/content/X3dTooltips.html#BooleanTrigger

IntegerSequencer node

IntegerSequencer outputs a series of SFInt32 integer values, based on a floating-point fractional input and *key, keyValue* pairs

- Continuous input similar to interpolator nodes
- Discrete output similar to BooleanSequencer

As with interpolator nodes, output field is named *value_changed*

Useful fields: node triggering via boolean events

- SFBool inputOnly *next:* send next *keyValue*
- SFBool inputOnly *previous:* send prior *keyValue*
- Provides a helpful alternative to using the more typical TimeSensor *fraction_changed*

Be sure to note both the similarities and differences between the ScalarInterpolator and IntegerSequencer nodes.

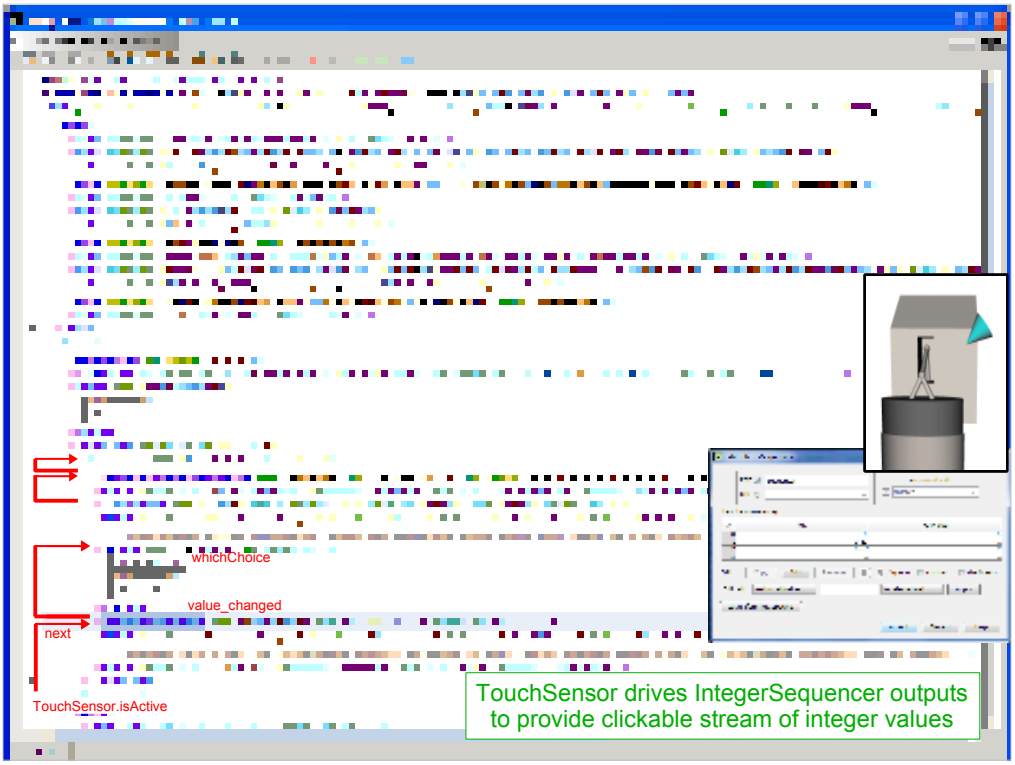
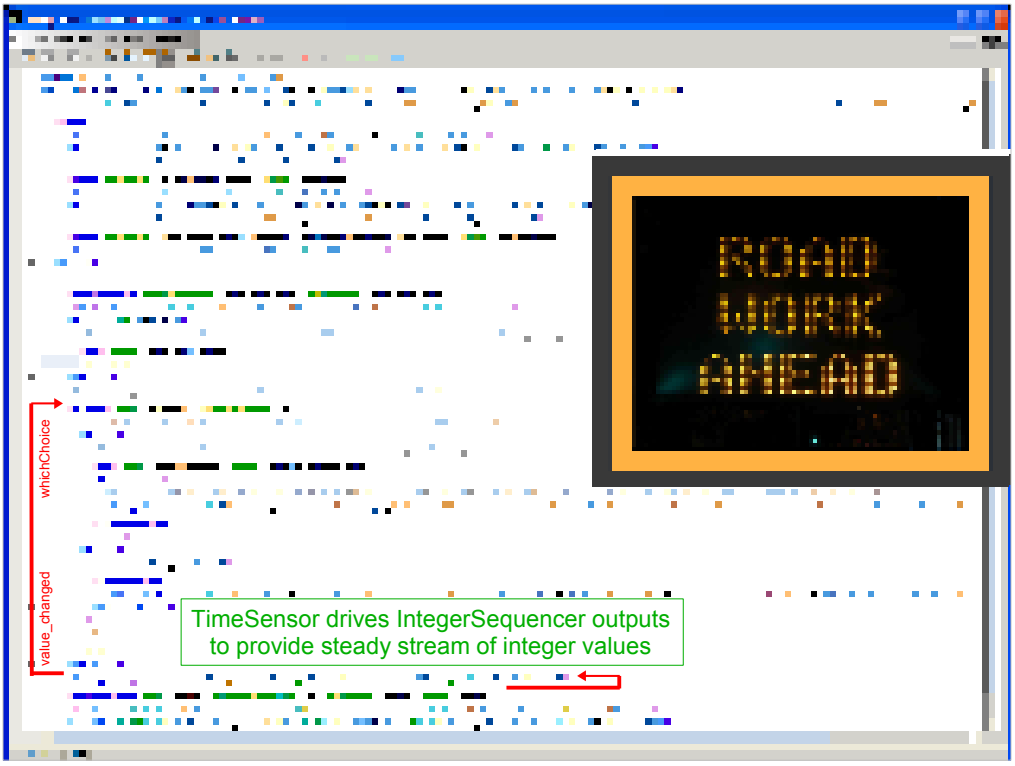


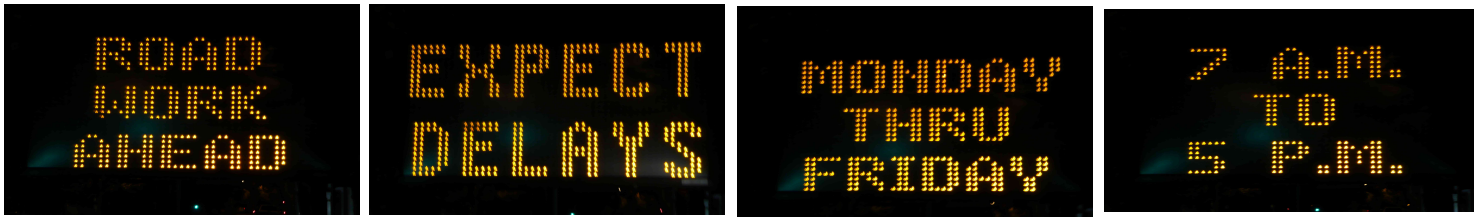
Figure 9.7, page 263, *X3D for Web Authors*

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/IntegerSequencerPumpHouse.x3d



http://x3dGraphics.com/examples/X3dForWebAuthors/Chapter07-EventAnimationInterpolation/IntegerSequencerRoadSignSwitcher.x3d

These images were taken with a simple digital camera January 2009 on Pacific Street in Monterey California. Rudimentary cropping and rotation was performed using GIMP image-processing software. Note that initial and final images are identical to permit smooth looping.

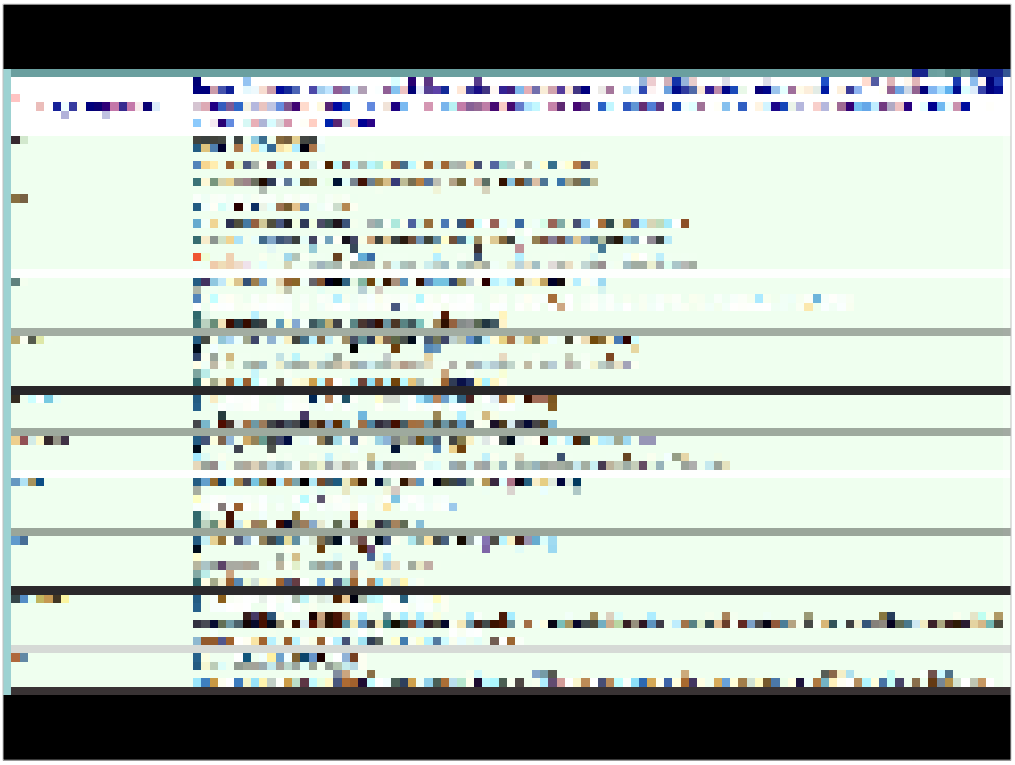


X3D Scene Authoring Hints – Images

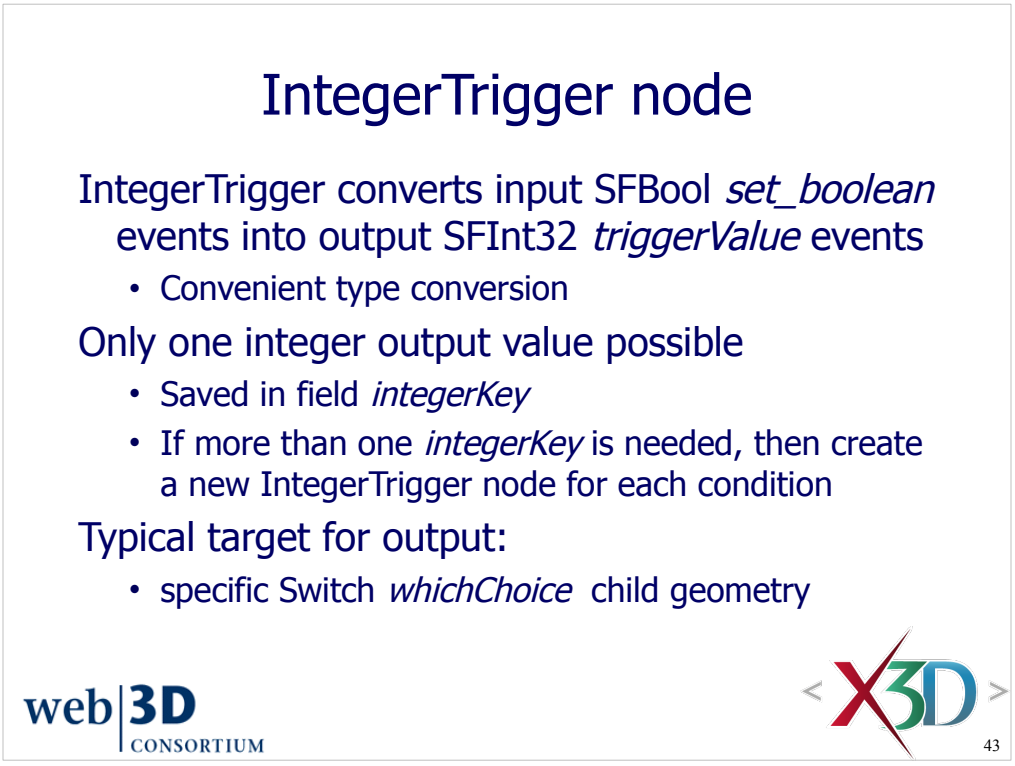
- http://www.web3d.org/x3d/content/examples/X3dSceneAuthoringHints.html#Images
- An excellent open-source image-editing program is the GNU Image Manipulation Program (GIMP). Runs on multiple operating systems. http://www.gimp.org

Be careful out there. Some further images, for fun (or the coming zombie apocalypse):

- http://images.google.com/images?q=danger%20zombies%20ahead



http://www.web3d.org/x3d/content/X3dTooltips.html#IntegerSequencer



Because IntegerTrigger only holds a single integerKey value, usually the author needs to define and connect multiple IntegerTrigger nodes to support multiple selections. This is OK because independent routing logic is needed for each selection anyway.

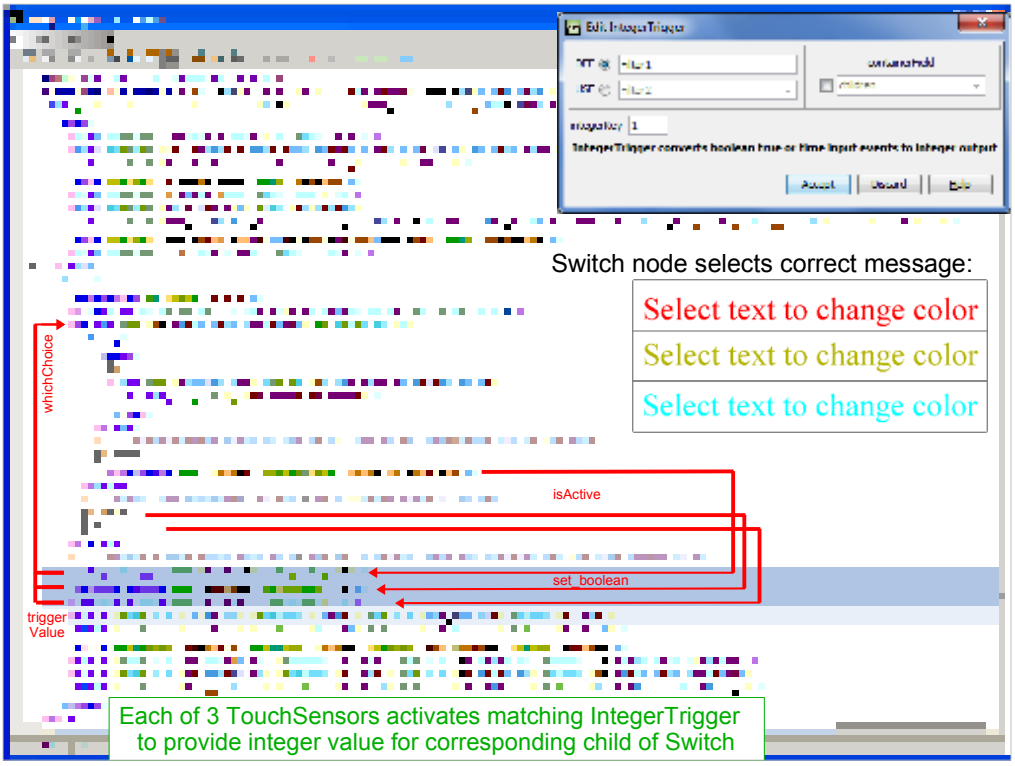
From X3D specification 30.4.6 IntegerTrigger:

> IntegerTrigger handles single field Boolean events to set an integer value for the output event. This is useful for connecting environmental events to the Switch node's whichChoice field.

Upon receiving a set_boolean event, the IntegerTrigger node will generate a triggerValue event with the current value of integerKey.

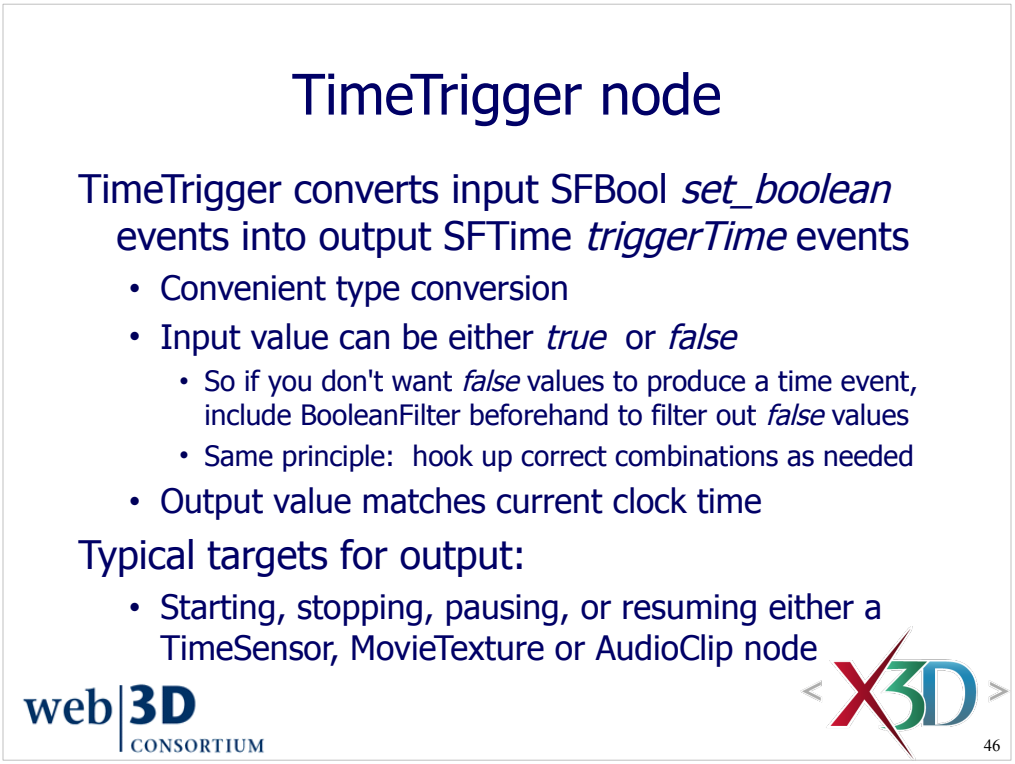
Node comparison:

- BooleanTrigger (time → boolean)
- IntegerTrigger (boolean → integer)
- TimeTrigger (boolean → time)



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/IntegerTrigger.x3d

http://www.web3d.org/x3d/content/X3dTooltips.html#IntegerTrigger

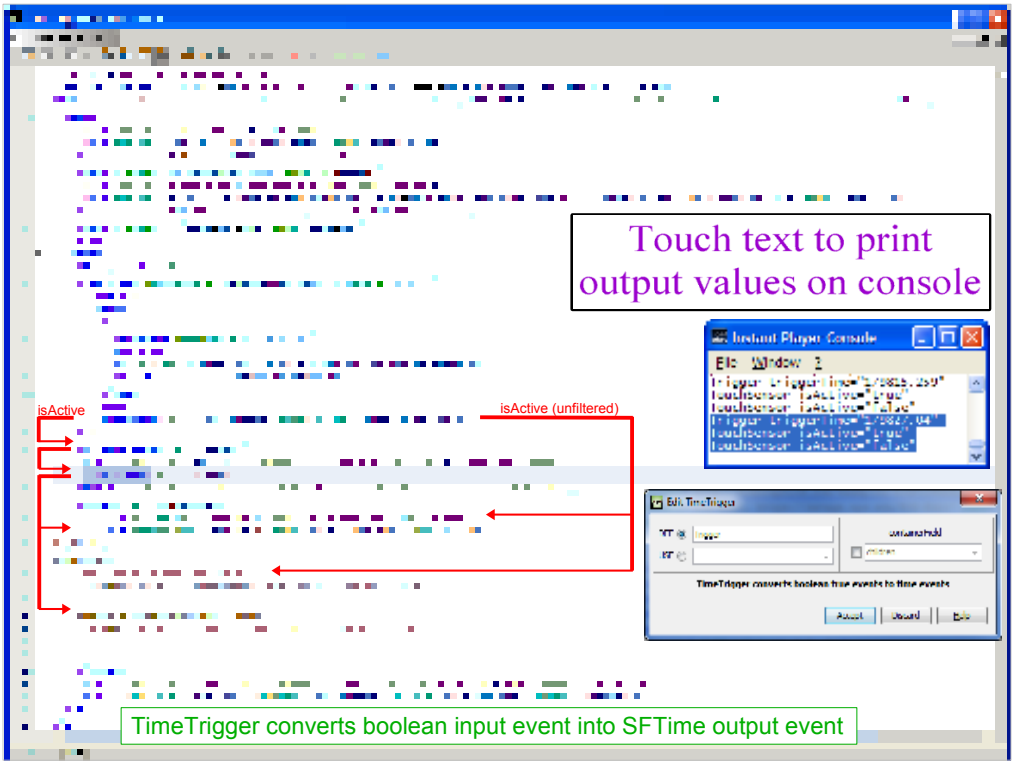


From X3D specification 30.4.7 TimeTrigger:

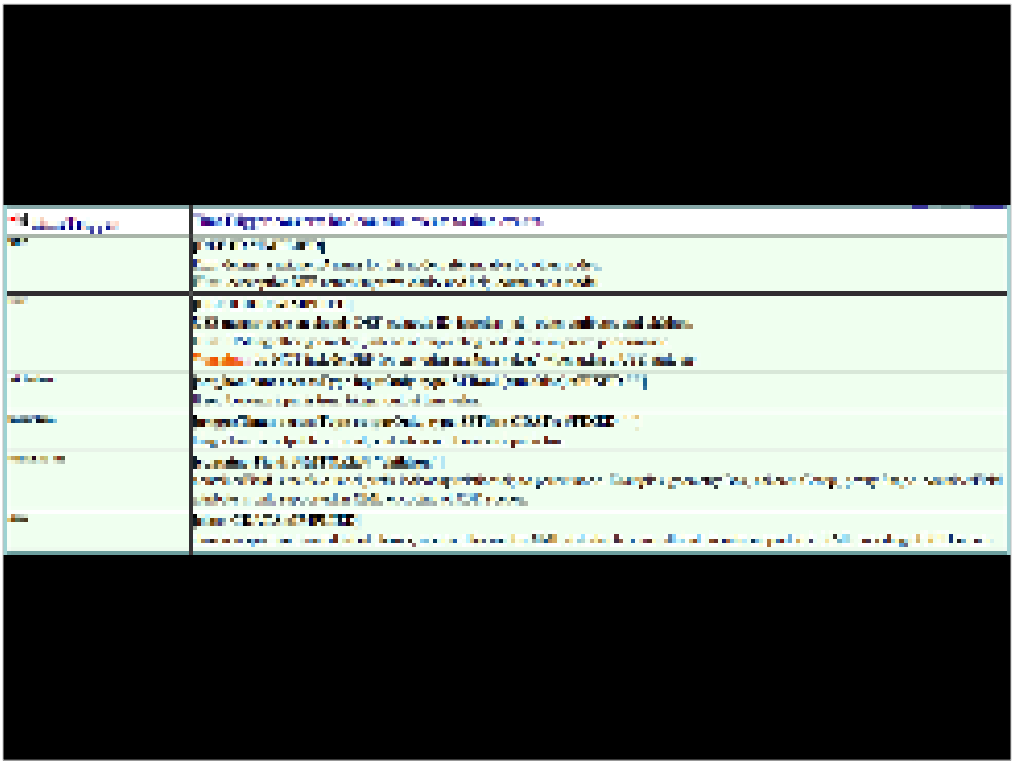
> TimeTrigger is a trigger node that generates time events upon receiving Boolean events. The triggerTime event is generated when the TimeTrigger receives a set_boolean event. The value of triggerTime shall be the time at which set_boolean is received. The value of set_boolean shall be ignored.

Node comparison:

- BooleanTrigger (time → boolean)
- IntegerTrigger (boolean → integer)
- TimeTrigger (boolean → time)



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/TimeTriggerTest.x3d



http://www.web3d.org/x3d/content/X3dTooltips.html#TimeTrigger

Script node

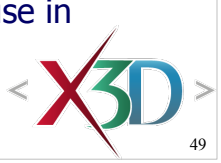
Script encapsulates programmatic source code

- Execution is triggered via arrival of input event
  - Each input field (i.e. method) receives a value, ← Declarative
  - performs algorithmic calculations, then ← Imperative
  - can return response event(s) via output fields ← Declarative

This approach preserves X3D event model for animation, allowing Script code within the scene graph to perform complex computations

- In response to ROUTEd scene-graph event inputs
- Similarly can return typed X3D values for use in scene graph via ROUTE connections

web3D CONSORTIUM



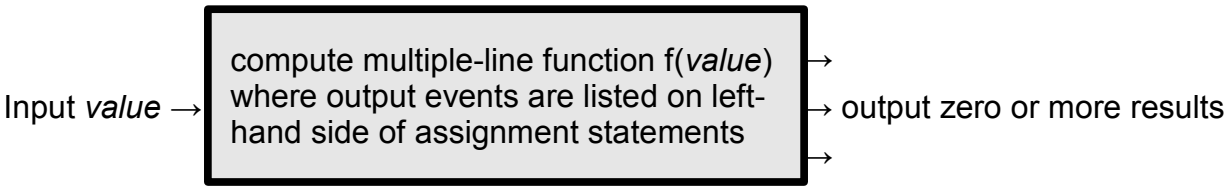
49

The Script node is the second half of this chapter. Script is not an event utility node.

The description in this slide shows that the purpose of the Script node is to look like other nodes in the X3D scene graph, acting and reacting in a typical manner by sending and receiving events. Being hooked up via ROUTE connections to send and receive events is clearly declarative. Internally the script code itself encapsulates step-by-step imperative algorithms.

Each input-field definition in the Script approach has a corresponding method to do the necessary computational work. Thus each can be thought of as a simple function, providing cause-and-effect response to any inputs received.

Functional (black box) model:



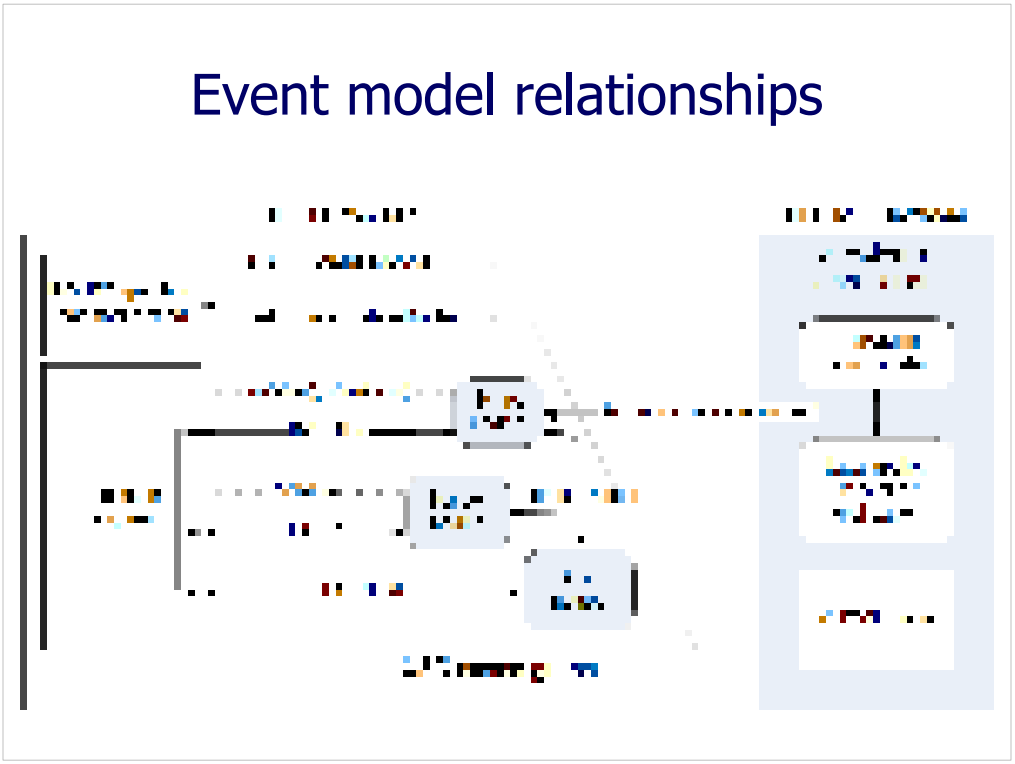


Figure 9.11, page 272, *X3D for Web Authors*.

Note that this diagram is a general description of architectural components and relationships within an X3D browser. The details of actual browser software may vary quite a lot, since they are allowed to decide how to actually implement the combined functionality of X3D.

Thus authors don't need to worry too much about the fine points of this diagram. It simply describes how things are likely to work “under the hood” of the browser engine.

Simple X3D execution model

- Display image from double buffer
- Advance and freeze the simulation clock
- Generate and process all events
  - Then stop, don't loop indefinitely!
- Draw new image in back buffer
- Swap buffer, replace with new image

web|3D CONSORTIUM

51

X3D execution model

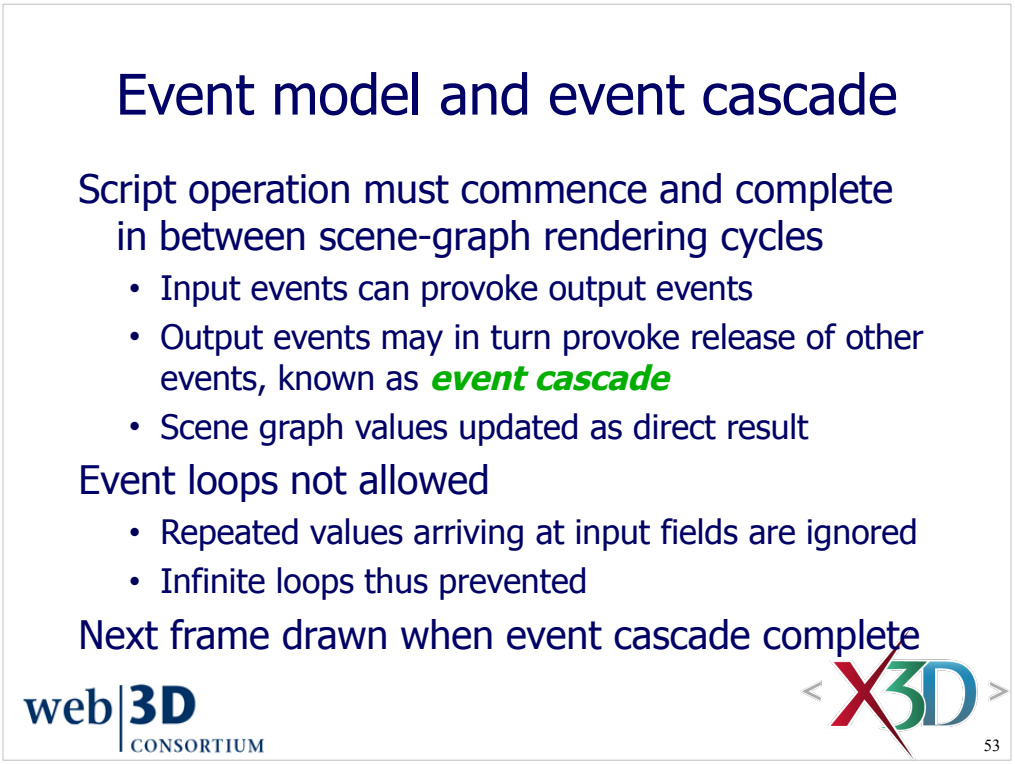
0. Prior image frame is already displayed on screen
1. Update camera based on currently bound Viewpoint
2. Evaluate inputs from sensors and event producers, then place events in queue for delivery
3. ROUTE events to defined destinations, updating scene-graph fields as appropriate
4. If any events generated as a result of steps 2 & 3, repeat until all events delivered in event cascade
5. Pixels in next screen update are rendered to image
6. Double-buffer display swaps new, old frame images
7. Browser's simulation clock is updated to match system's real-time clock (advancing 'one tick')
8. Single loop done, repeat from step 1 indefinitely

The X3D execution model describes the sequence of steps needed update the view of an X3D scene: react to user inputs and ROUTE values, compute any updated values, and render a new image for screen display.

Rephrase: “how do X3D players draw screen updates for your scene?” This is how.

Question: why don't we draw straight to the screen, instead of an off-screen image?

Answer: because the user might see ghostly partial/incorrect images while the new frame is being drawn.



This functional sequence of steps is common to many different 3D-graphics rendering programs and interface libraries.

field declarations

Each Script can contain <field> definitions

- Which are the interface for the Script
- Zero or more <field> definitions allowed

<field> definitions must be exactly correct

- Define *name*, *type*, *accessType*, and initial *value*
- SFNode, MFNode initializations are contained
- *accessType* initializeOnly, inputOnly must have an initial *value*
- inputOnly, outputOnly have no initial *value*

web3D CONSORTIUM

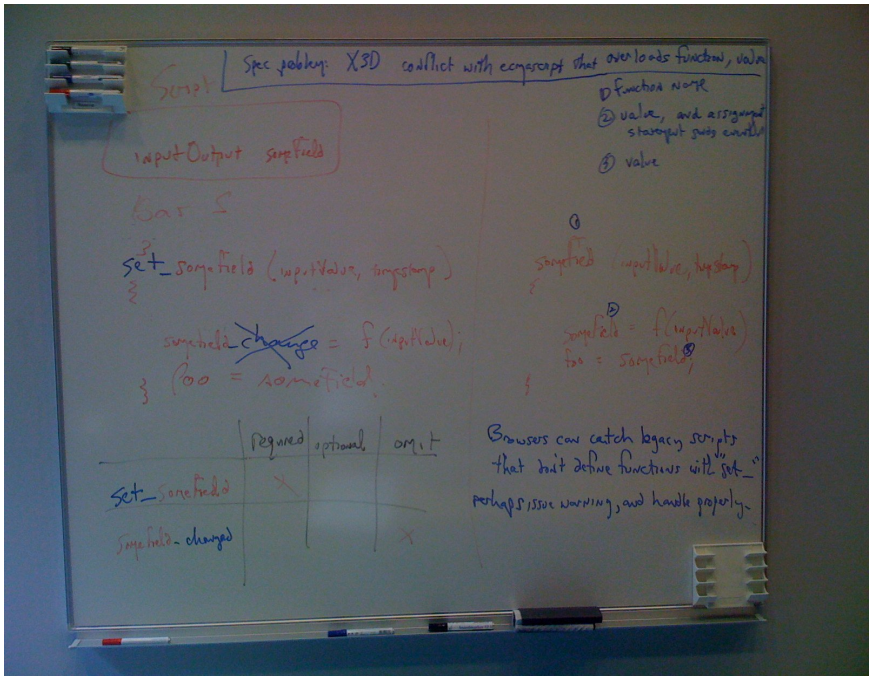
<X3D>

54

Corresponding ClassicVRML construct: [square brackets around field definitions] as shown in Table 14.2, pp. 386-387.

Since authors define inputs, outputs and functional characteristics of Script nodes, they can be considered another extensibility mechanism for the X3D language.

A lot of careful thought and practice has gone into this design. Here is a diagram produced at the Web3D 2009 Symposium in Darmstadt Germany on method naming.



Field-Type Names	Description	Default Values
SFBool	Single-Field boolean value	false (XML syntax) or FALSE (ClassicVRML syntax)
MFBool	Multiple-Field boolean array	Empty list
SFColor	Single-Field color value, RGB	0 0 0
MFColor	Multiple-Field color array, RGB	Empty list
SFColorRGBA	Single-Field color value, red-green-blue alpha (opacity)	0 0 0 0
MFColorRGBA	Multiple-Field color array, red-green-blue alpha (opacity)	Empty list
SFInt32	Single-Field 32-bit integer value	0
MFInt32	Multiple-Field 32-bit integer array	Empty list
SFFloat	Single-Field single-precision floating-point value	0.0
MFFloat	Multiple-Field single-precision floating-point array	Empty list
SFDouble	Single-Field double-precision floating-point value	0.0
MFDouble	Multiple-Field double-precision array	Empty list
SFImage	Single-Field image value	0 0 0 Contains special pixel-encoding values, see Chapter 5 for details

Table 14.3, page 388, *X3D Field Types and Default Values*

Since all field values in X3D nodes are strongly typed, the fields you define in a Script node must also be typed.

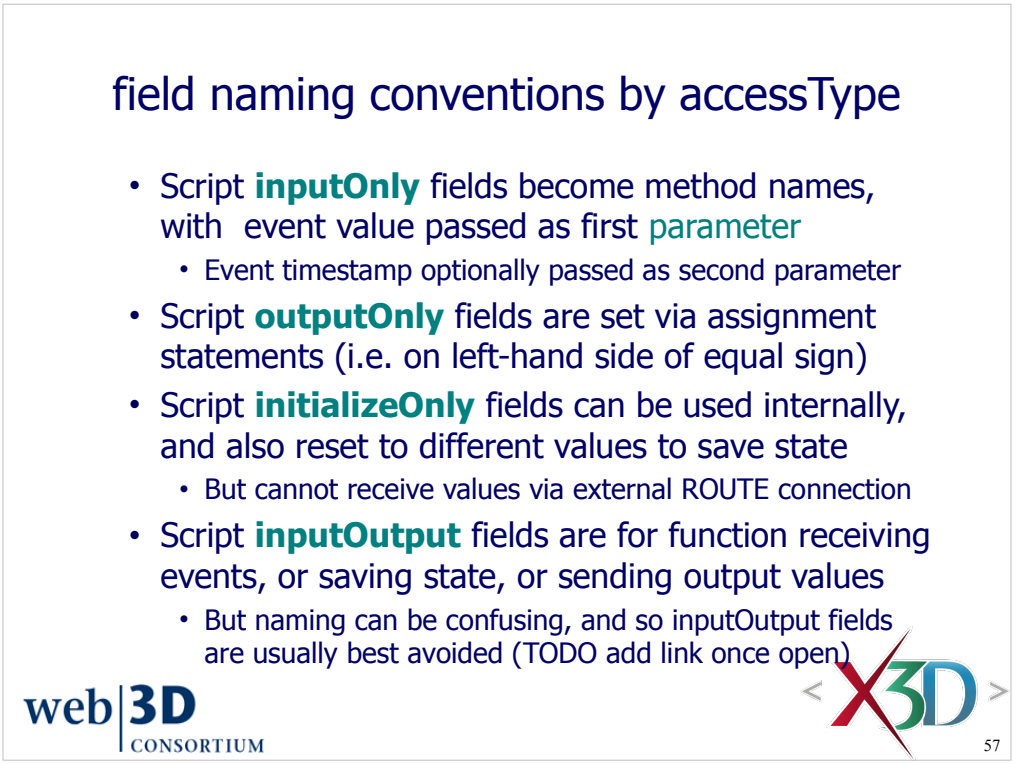
There is no mechanism to define new data types in X3D. This makes sense – even if you were able to create a new type, none of the existing nodes would be able to use it. Nevertheless, since arrays can be constructed for strings, booleans, integers and floating-point types, authors can typically figure out what they really need for any job at hand.

Rephrased using “hardware-store” logic:

- if you don't find the X3D type you want, you probably don't need it!

MFImage	Multiple-Field image value	Empty list
SFNode	Single-Field node	Empty node, NULL
MFNode	Multiple-Field node array of peers	Empty list
SFRotation	Single-Field rotation value using 3-tuple axis, radian-angle form	0 0 1 0
MFRotation	Multiple-Field rotation array	Empty list
SFString	Single-Field string value	Empty string, representable as two adjacent quotation marks
MFString	Multiple-Field string array	Empty list
SFTime	Single-Field time value	–1, sentinel indicating no time value.
MFTime	Multiple-Field time array	Empty list
SFVec2f/SFVec2d	Single-Field 2-float/2-double vector value	0 0
MFVec2f/MFVec2d	Multiple-Field 2-float/2-double vector array	Empty list
SFVec3f/SFVec3d	Single-Field vector value of 3-float/3-double values	0 0 0
MFVec3f/MFVec3d	Multiple-Field vector array of 3-float/3-double values	Empty list

Table 14.3, page 388, *X3D Field Types and Default Values*



Similar construct: [in, out] parameter notation in CORBA's Interface Definition Language (IDL). Related (but not identical) references:

- Wikipedia: Interface description language
- http://en.wikipedia.org/wiki/Interface_definition_language
- OMG Tutorial on OMG IDL
- http://www.omg.org/gettingstarted/omg_idl.htm
- OMG Specification of OMG IDL
- http://www.omg.org/gettingstarted/omg_idl.htm

02-06-39 (CORBA 3.0 - OMG IDL Syntax and Semantics chapter)

3.13.2 Parameter Declarations

A parameter declaration must have a directional attribute that informs the communication service in both the client and the server of the direction in which the parameter is to be passed. The directional attributes are:

- **in** - the parameter is passed from client to server.
- **out** - the parameter is passed from server to client.
- **inout** - the parameter is passed in both directions.

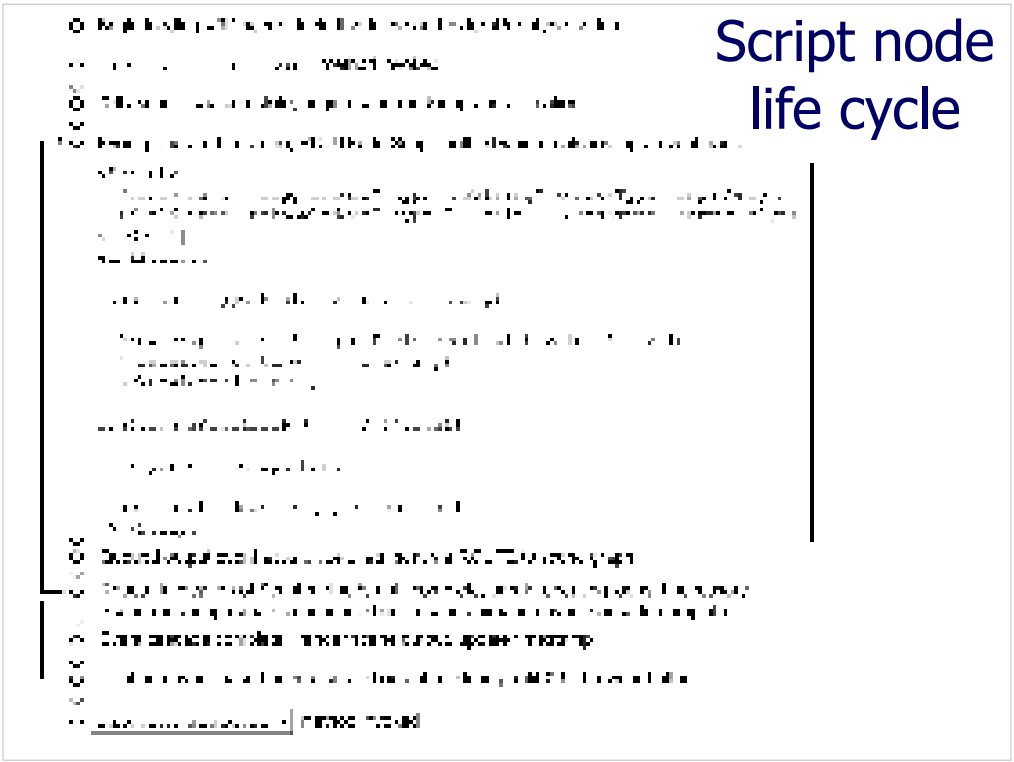


Figure 9.12, page 274, *X3D for Web Authors*.

The Script node life cycle describes the steps that occur from Script loading and initialization through run-time operation and eventual shutdown.

ECMAScript (JavaScript)

ECMAScript is primary language for X3D scripting

- Required for X3D players that support scripting
- Originally (and often) called JavaScript
- Renamed ECMAScript when standardized via European Computer Manufacturers Association http://www.ecma-international.org
- Scene Access Interface (SAI) defines standard interfaces for interaction with scene graph

Can be embedded within X3D scene, or kept separately in an external .js file

- X3D-Edit provides code colorization and syntax support when editing separate .js file

Although the term ECMAScript may seem a bit awkward at first, it is precisely correct. Essentially it is the same as Javascript. Note however that there are a variety of almost-but-not-quite-identical language definitions that call themselves JavaScript.

When verbally describing Script operations, it is further hard to tell what someone is saying when they discuss “my Java Script (node)” or “my JavaScript (code).” Since Java (and Java scripts) preceded JavaScript by several years, we might speculate that there is an especially confusing place in Hades for the folks who originally approved the name “JavaScript.”

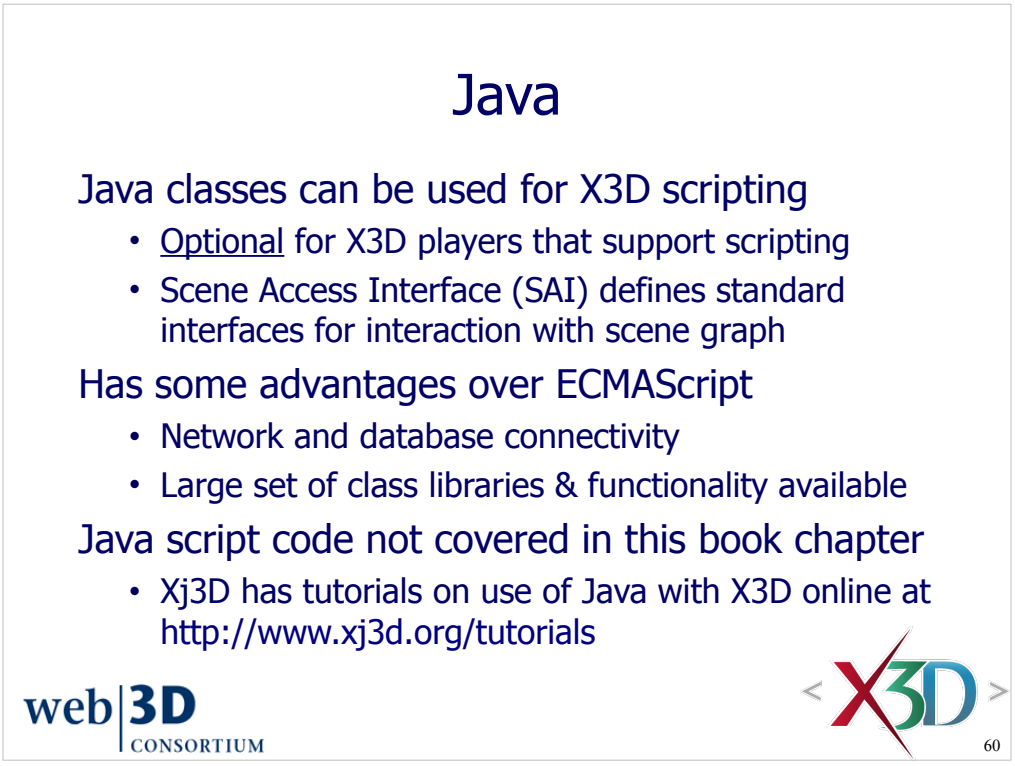
HTML authors also write scripts in Javascript.

Helpful references:

- http://en.wikipedia.org/wiki/ECMAScript
- http://en.wikipedia.org/wiki/JavaScript
- http://en.wikipedia.org/wiki/JavaScript_syntax

ECMAScript ECMA-262 Specification

- http://www.ecma-international.org/publications/standards/Ecma-262.htm
- http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf
- http://www.web3d.org/specifications/Ecma-262.pdf



Helpful references:

- http://java.sun.com Java developer page
- http://java.com Java in action, Java downloads
- http://java.com/en/about Learn about Java
- http://java.sun.com/new2java New to Java Programming
- http://en.wikipedia.org/wiki/Java_(programming_language)

X3D-Edit itself is developed using Netbeans, which is written in Java and provides rich support for Java.

We plan to add support for X3D scripting using Java to X3D-Edit. A start in that direction is provided by the Add New Java Sample Script Code button on the toolbar, which is also available under the File > New X3D menu.

- http://www.web3d.org/x3d/content/examples/newX3dScript.java

We hope to cover X3D scripting using Java in a future book.

Java generation example

Java can also be used for creation of X3D scenes

- Output from standalone programs
- Output from existing programs

Example program: CircleLines.java

- Invoked via command line
- User specifies number of points in circle (default 24)

```
Usage:   java CircleLines [numberOfSegments] [ > CircleLinesFileName.x3d ]
Example: java CircleLines 60 > CircleLines60.x3d
```

Example scene results:

- CircleLinesExample.x3d
- CircleLinesExample60.x3d

web3D CONSORTIUM

<X3D>

61

Helpful references:

- http://java.sun.com Java developer page
- http://java.com Java in action, Java downloads
- http://java.com/en/about Learn about Java
- http://java.sun.com/new2java New to Java Programming
- http://en.wikipedia.org/wiki/Java_(programming_language)

Java

X3D-Edit itself is developed using Netbeans, which is written in Java and provides rich support for Java.

NetBeans

We plan to add support for X3D scripting using Java to X3D-Edit. A start in that direction is provided by the Add New Java Sample Script Code button on the toolbar, which is also available under the File > New X3D menu.

- http://www.web3d.org/x3d/content/examples/newX3dScript.java

We hope to cover X3D scripting using Java in a future book.

url field and CDATA block

Local and online url addresses for script code contained as elements of url string array

- Essentially same approach as other url fields
- Each address intended to point to a copy of same functionally equivalent scripting code

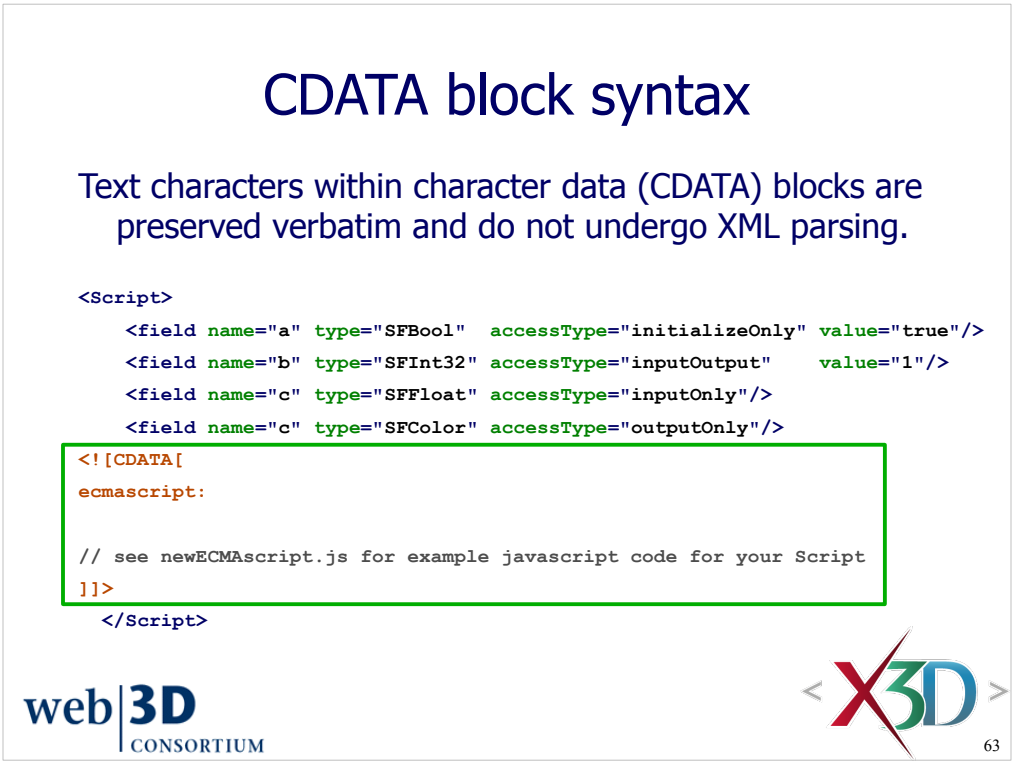
One url value can also contain `ecmascript:` code

- This approach is used in ClassicVRML encoding

Preferred X3D approach for XML encoding is to encapsulate in CDATA block, which prevents unintended XML parsing of character data

CDATA character data => "hey XML leave it alone!!"

CDATA "character data" blocks protect contained text from being interpreted as XML by the XML parser. It is possible to avoid them by escaping some characters as character entities (for example, converting less-than sign **<** to **<**) but this approach degrades readability and makes the code less portable.

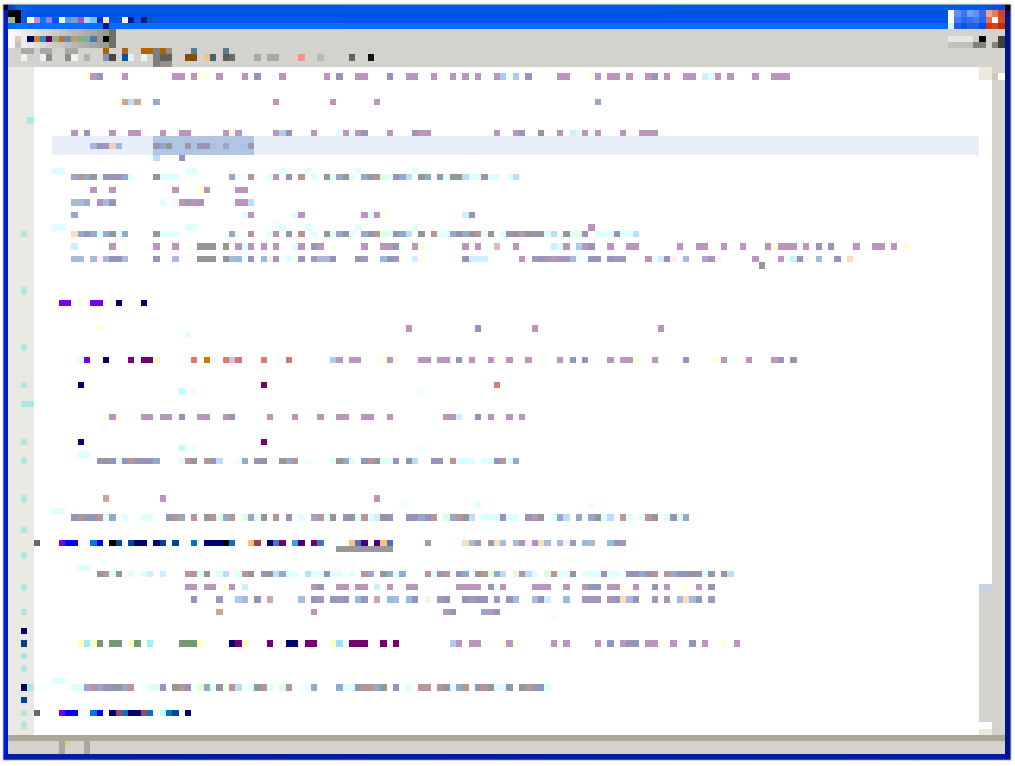


The contained CDATA syntax must be followed exactly. X3D-Edit facilitates this task.

CDATA blocks only appear within Script nodes, following any <field/> definitions. Do not use them anywhere else within an X3D scene.

CDATA blocks are not needed if ECMAScript source is in a separate external .js file.

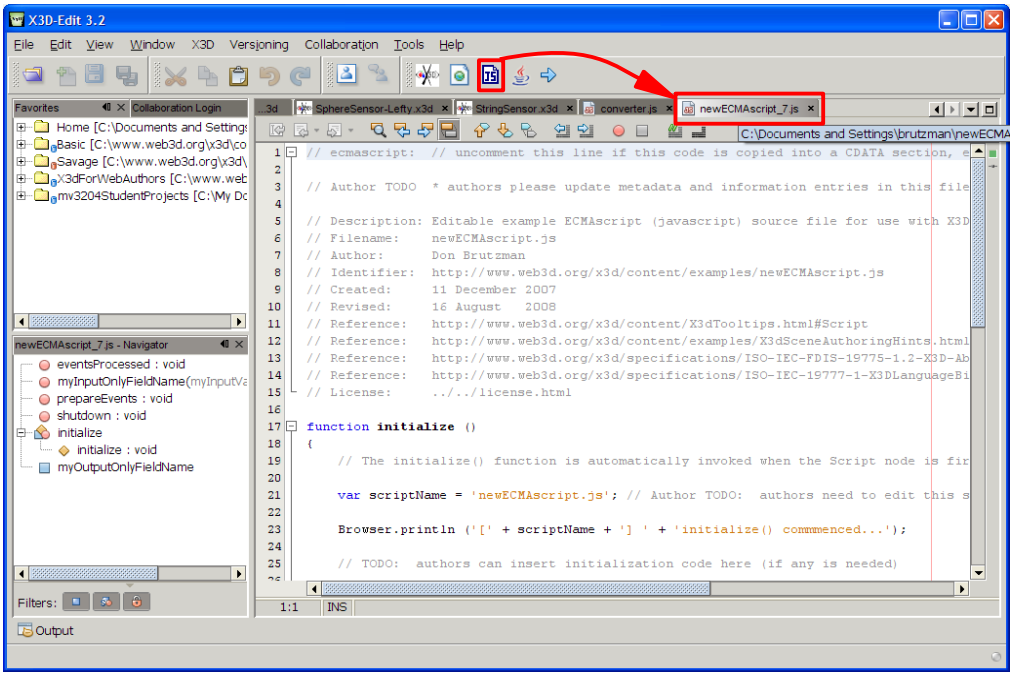
Note once again that only field definitions with *accessType*=”initializeOnly” or *accessType*=”inputOutput” receive an initialization value.



newECMAscript.js is a template for creating new scripts. You must rename and edit it.

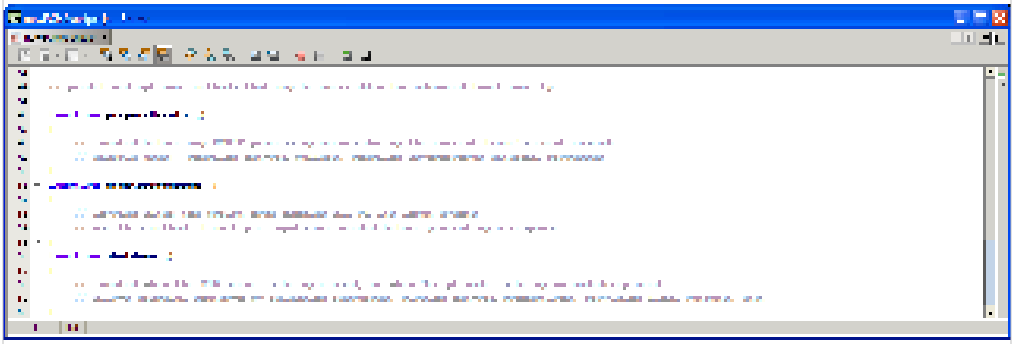
http://www.web3d.org/x3d/content/examples/newECMAscript.js

X3D-Edit provides a new ECMAscript.js file by selecting the menu *File* > *New X3D* > *NewECMAscript sample Script code.* Alternatively create one using the JS button:



newECMAscript.js part 2

... continued from previous slide:



Worth noting: these methods are rarely used

web3D CONSORTIUM

<X3D>

65

newECMAscript.js is a template for creating new scripts. You must rename and edit it.

http://www.web3d.org/x3d/content/examples/newECMAscript.js

<Script> and <field> editors

These are the X3D-Edit panes for editing <Script> and <field> definitions. Note that an entry box for a simple-type field initialization value is only offered when appropriate.

Example: ScriptSimpleStateEvents.x3d

Pushbutton switch changes lamp color

- TouchSensor detects button select, acts as trigger
- TimeSensor animates pushbutton cylinder
- Script invoked, checks current state to decide logic
  - If button now down, color lamp yellow and reset path
  - If button now up, color lamp grey and reset path

Script code for this example kept in external file

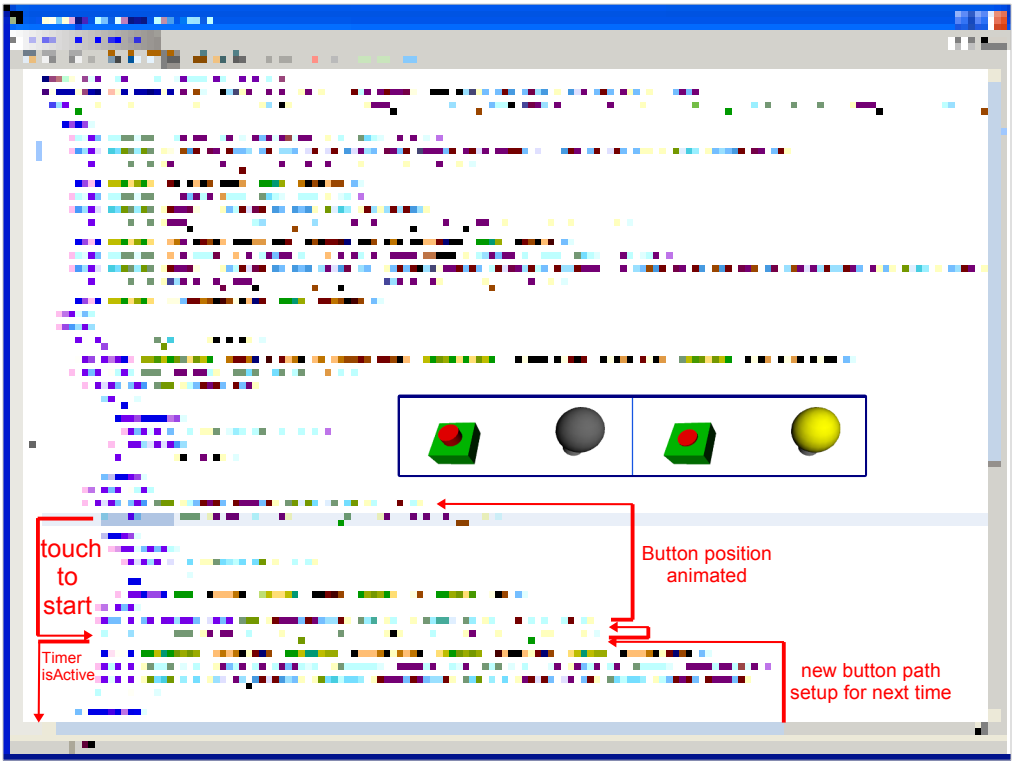
- Still has access to all defined fields from within code
- Don't include `ecmascript:` header in external file

X3D event model still governs animation flow

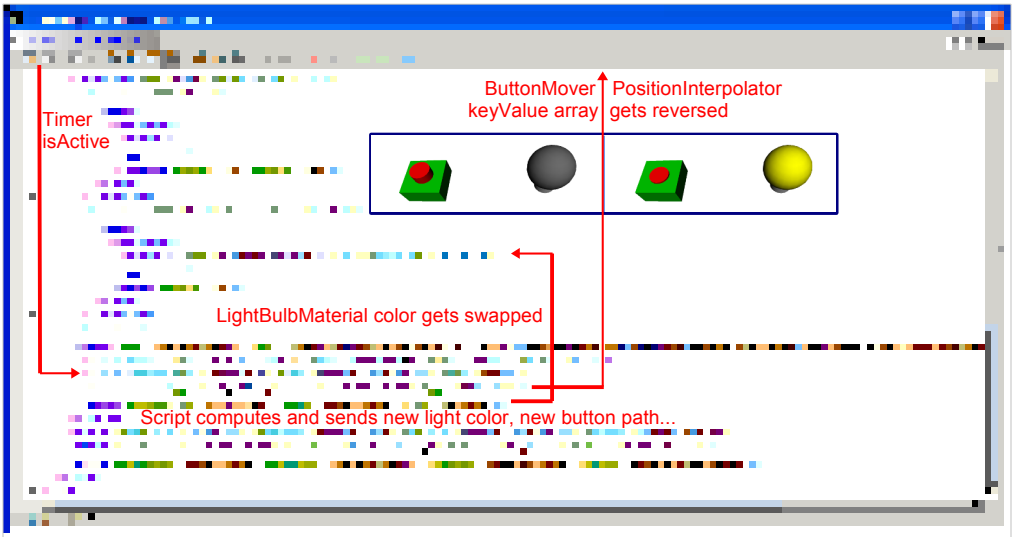
web3D CONSORTIUM

67

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.x3d



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.x3d



Script located in external ScriptSimpleStateEvents.js file

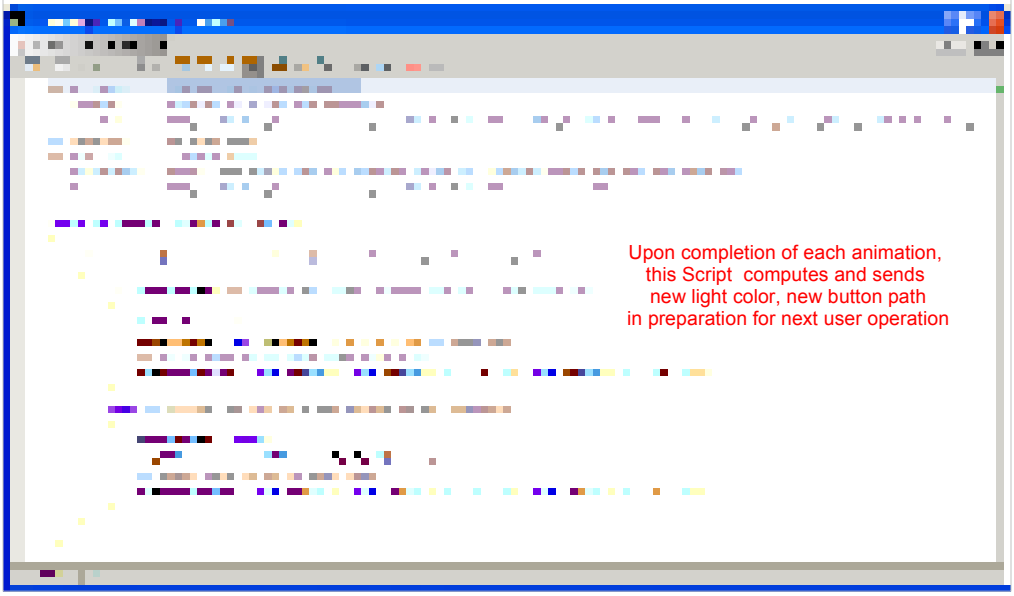
- buttonTimerActive function name matches inputOnly event; value contains passed event
- outputOnly events set by assignment statements

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.x3d

Note that both local and online addresses are provided in the Script *url* field in order to improve reliability that the ECMAScript code is found.

```
<Script DEF='ControlScript' url='"ScriptSimpleStateEvents.js"
"http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.js"'>
```

ScriptSimpleStateEvents.js



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptSimpleStateEvents.js

Example: ScriptEvents.x3d

Clicking pump house starts cone animation

- TouchSensor detects pump select, acts as trigger
- TimeSensor drives Script response
- Script invoked, computes rotation and translations

Script code for this example kept within .x3d file

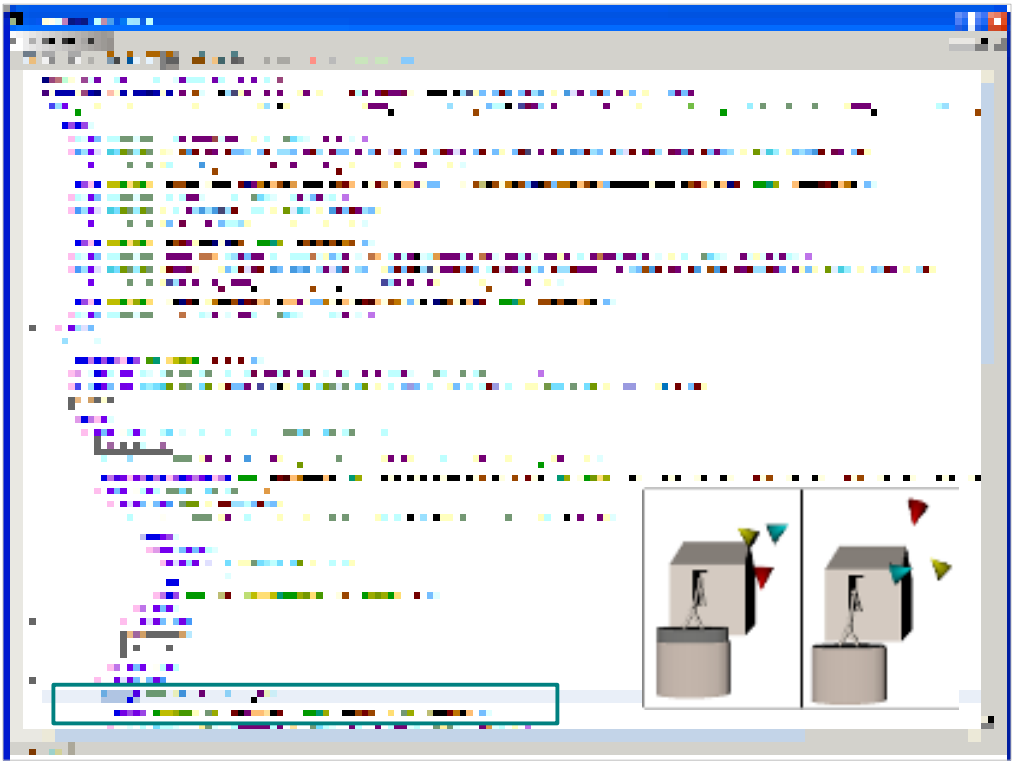
- Still has access to all defined fields from within code
- Must include `ecmascript:` header when internal
- CDATA block ensures that original text is preserved

X3D event model still governs animation flow

web|3D CONSORTIUM

71

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptEvents.x3d

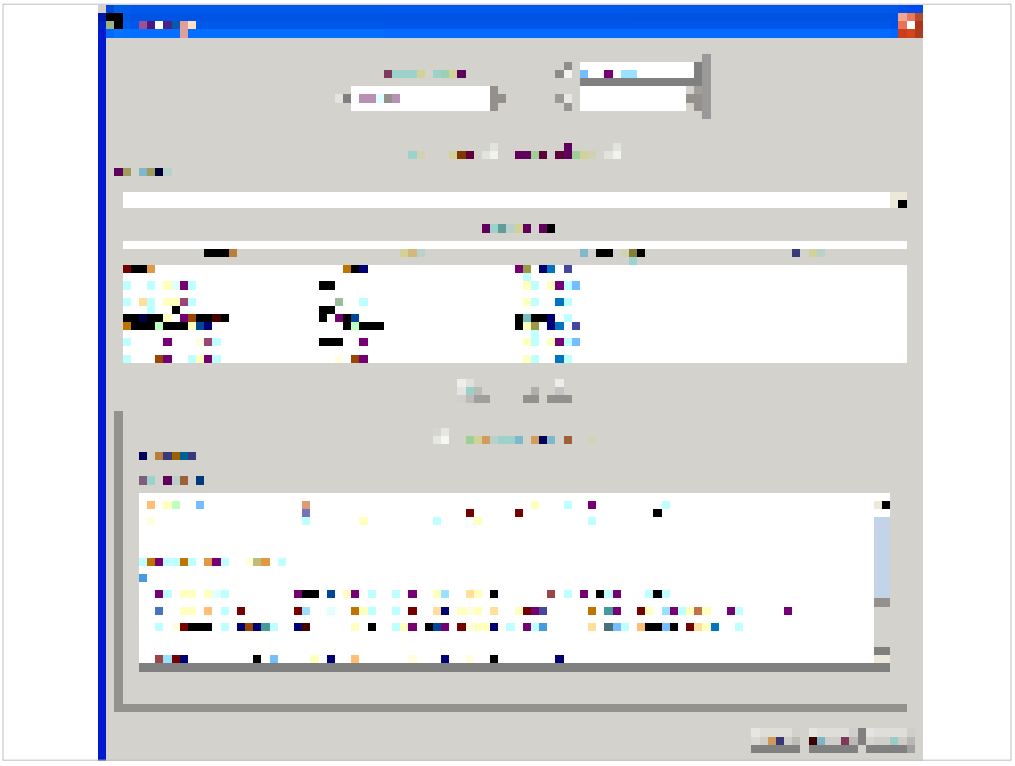


Figures 9.9 and 9.10, pages 267-268, *X3D for Web Authors*.

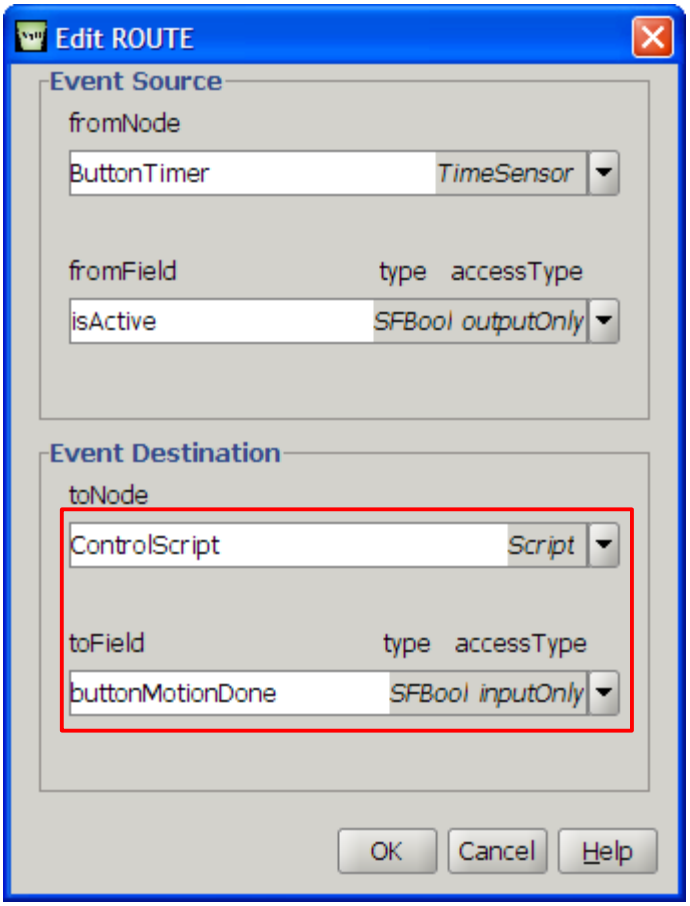
http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptEvents.x3d

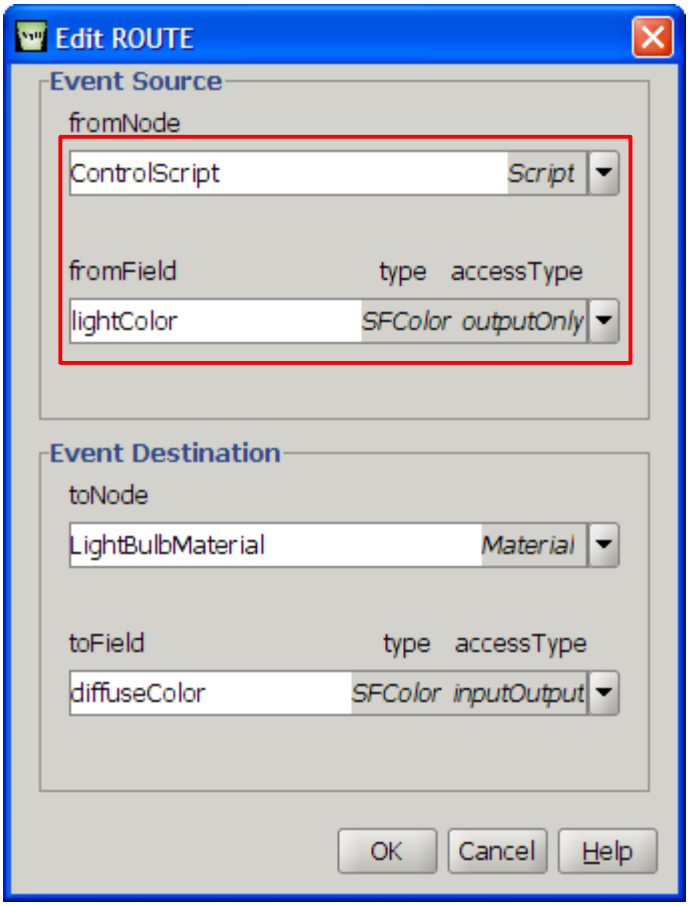
Figures 9.9 and 9.10, pages 267-268, *X3D for Web Authors*.

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptEvents.x3d



Note that the ROUTE editor also supports finding the appropriate input and output fields that you create for the Script. Examples from this scene:





Example: ScriptComplexStateEvents.x3d

Pushbutton switch changes 3-way lamp color

- TouchSensor detects button select, acts as trigger
- TimeSensor animates pushbutton cylinder
- Script invoked, checks current state to decide logic
  - Adds one to count each time, modifies color
  - Resets and turns off lamp once maximum reached

Script code kept in external file

- Still has access to all defined fields from within code
- Don't include `ecmascript:` header when external

X3D event model still governs animation flow

web3D CONSORTIUM

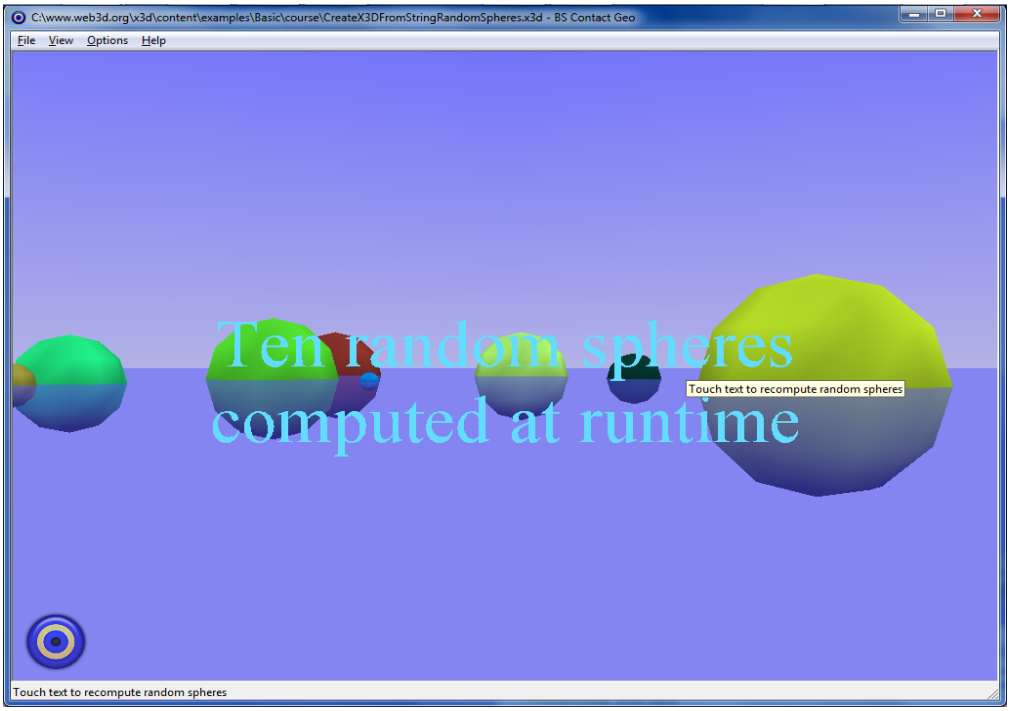
75

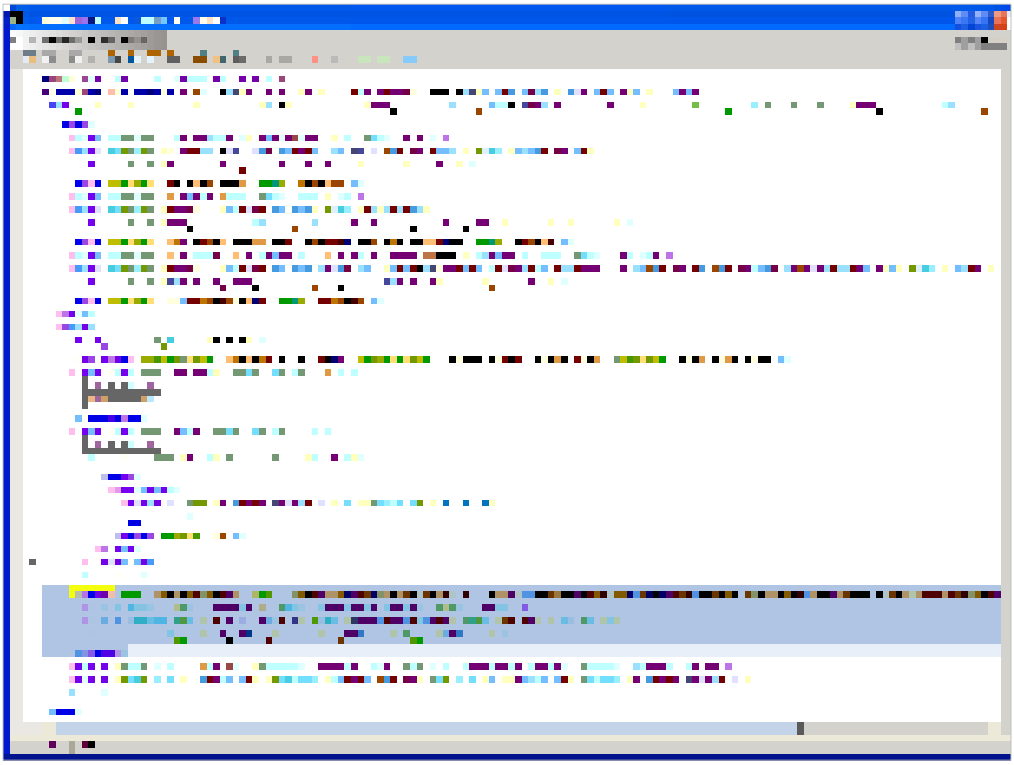
http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptComplexStateEvents.x3d

Interesting observation: probabilistic variables can be modeled by using a random number generation function and embedding a probability distribution function (pdf).

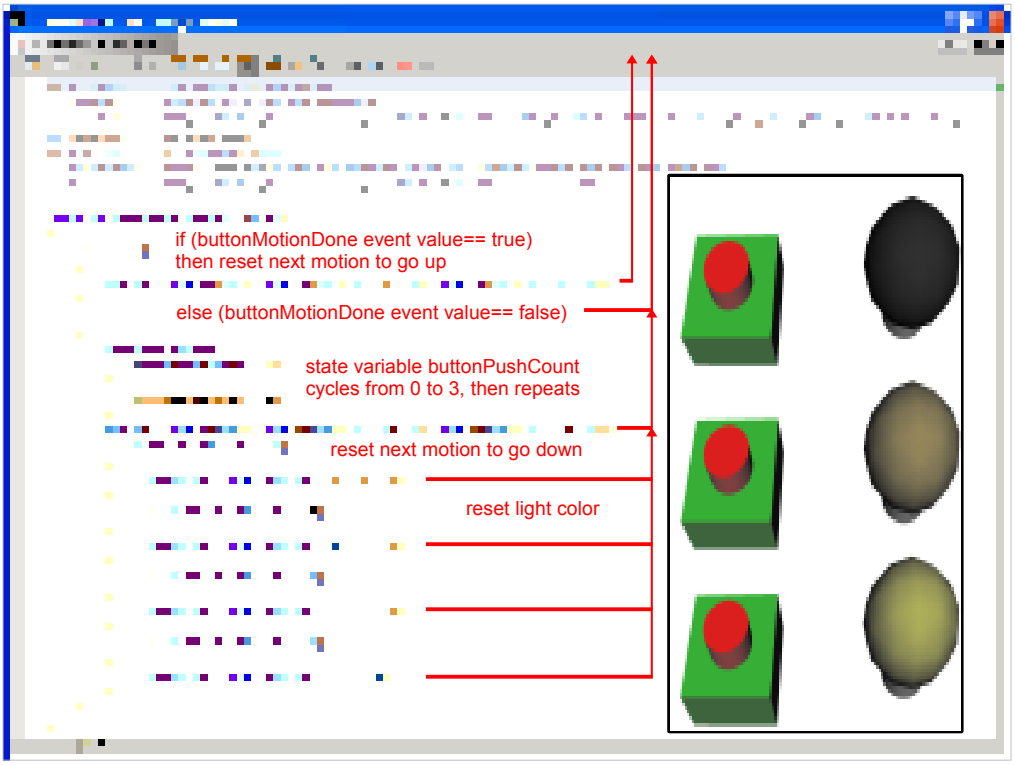
Here is a good example showing the use of random numbers from Basic X3D archive:

CreateX3DFromStringRandomSpheres.x3d



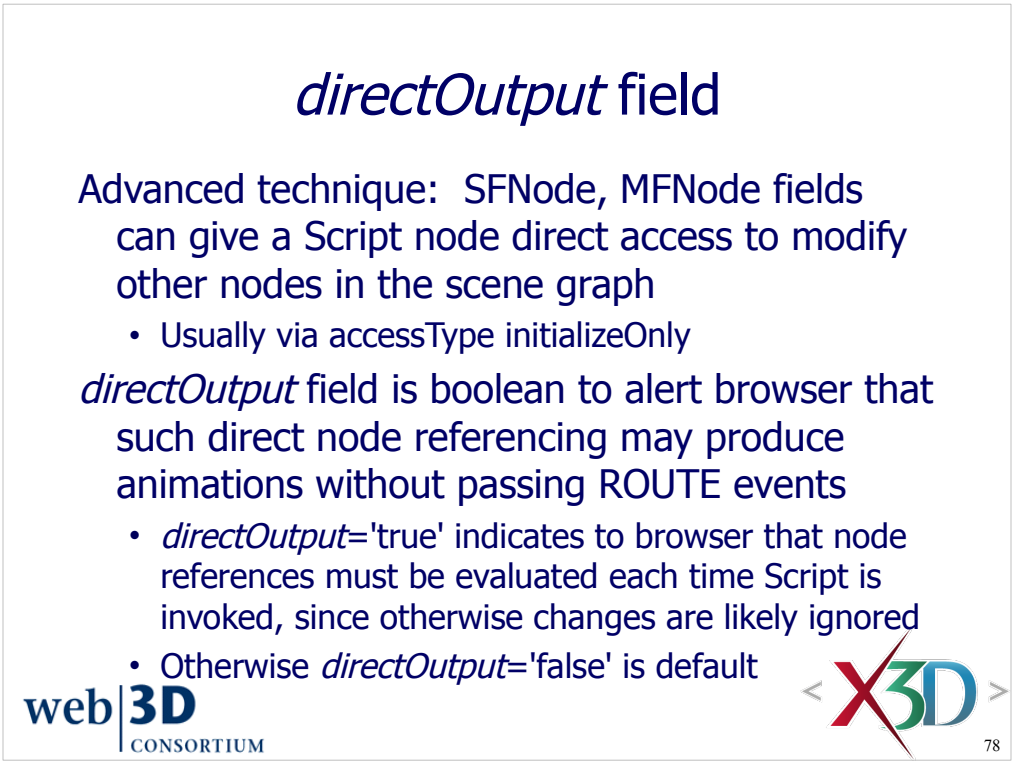


http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptComplexStateEvents.x3d



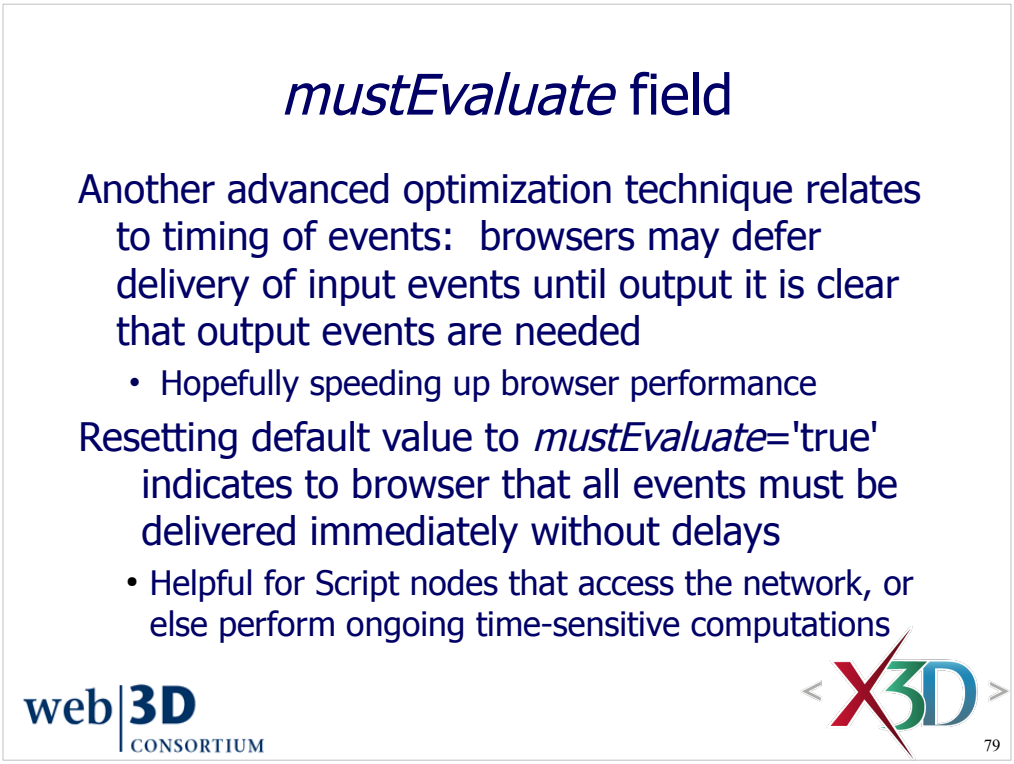
Figures 9.13, page 277, *X3D for Web Authors*.

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/ScriptComplexStateEvents.js

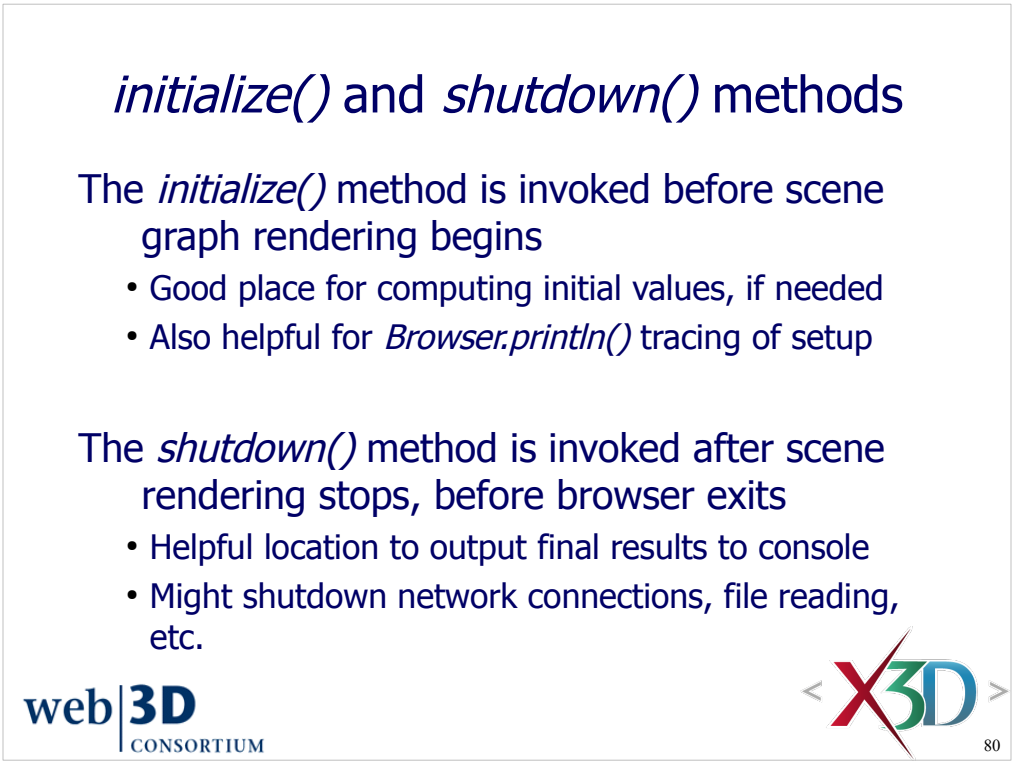


If a Script node includes an SFNode or MFNode field with *accessType*='initializeOnly' then it probably also needs to have *directOutput*='true' as well.

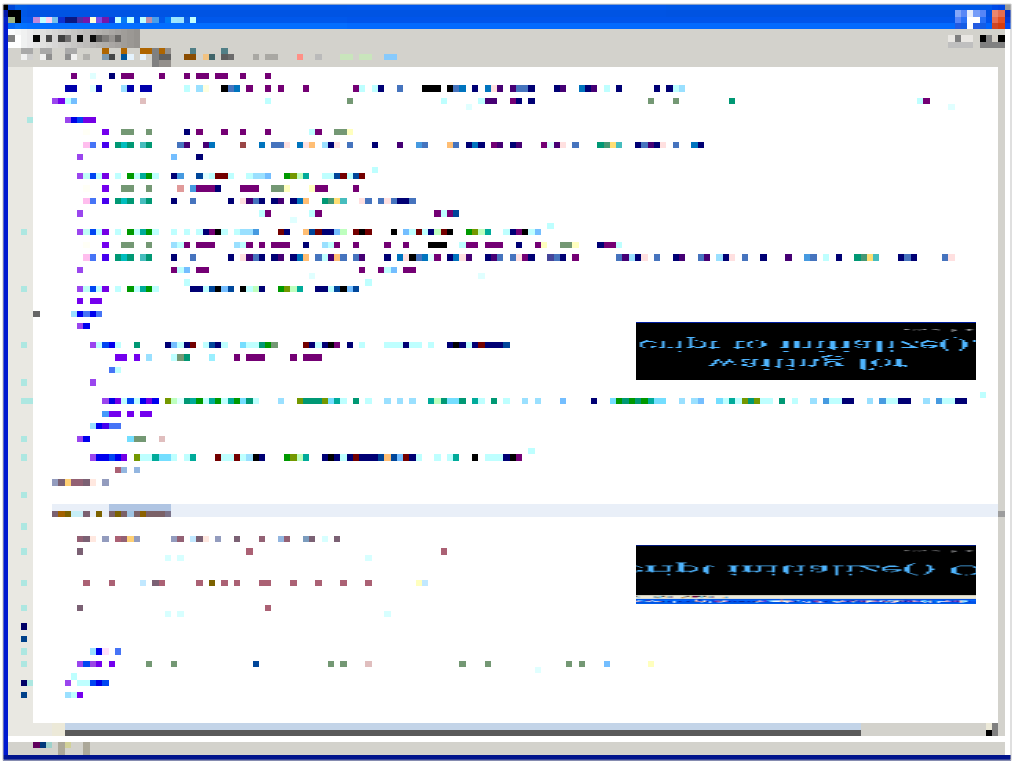
Thus *directOutput* provides an author hint to help with browser optimization.



The *mustEvaluate* field supports advanced techniques and is seldom overridden.



Note that *initialize()* method is frequently used, but *shutdown()* method is rarely used.



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter09-EventUtilitiesScripting/TestScriptInitialize.x3d

This scene tests proper operation of the `intialize()` method. The top snapshot is the default view of the scene, which will be displayed if scripting isn't working properly. The second snapshot is the expected result once the Script has initialized.

Note that output statements are also printed on the browser console:

```
[TestScriptInitialization.x3d initialize() commmenced...]
[...TestScriptInitialization.x3d initialize() complete]
```

Comparison of event and field control

ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeEventOutControl-EcmaScriptSnapshots.html

- Shows event control from *initialize()* to user interaction to *shutdown()*

ScriptNodeEventOutControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeFieldControl-EcmaScript.x3d

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeFieldControl-EcmaScriptSnapshots.html

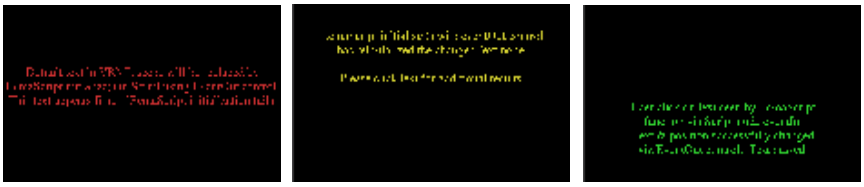
- Shows field control using passed node references, from *initialize()* to user interaction to *shutdown()*

Available via Basic Examples archive

- http://www.web3d.org/x3d/content/examples/Basic

web|3D CONSORTIUM

Screen snapshots of script operation



Visual operation is the same for each example

- Initial red text is the default scene setup
- Subsequent yellow text is immediately loaded during startup as a result of initialize() method
- Third set of green text is created by Script output, in response to user clicking yellow text

web3D CONSORTIUM

http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/ScriptNodeEventOutControl-EcmaScript.x3d

ROUTE control illustrated

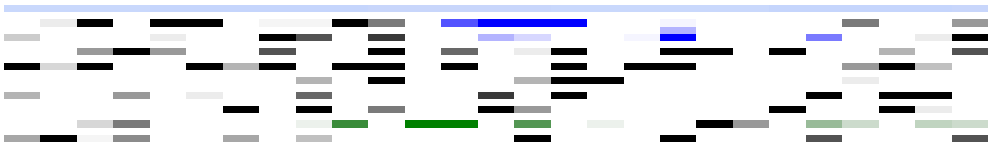


Illustration figures 4 and 5 excerpted from

- Brutzman, Don, “The Virtual Reality Modeling Language and Java,” *Communications of the ACM*, vol. 41 no. 6, June 1998, pp. 57-64.
- http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/VrmlJavaPaper.pdf

These figures illustrate direct event-based ROUTE control and indirect field-based control, respectively. The examples have been updated from VRML to X3D and retain the same functional structure.

Two sets of examples (using ECMAScript and Java) are provided for each approach.

TODO: proper testing of the revised Java examples using the X3D Scene Access Interface (SAI) interfaces.

Field control illustrated

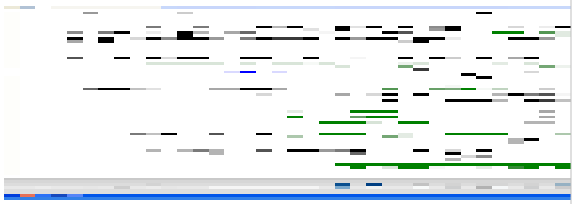


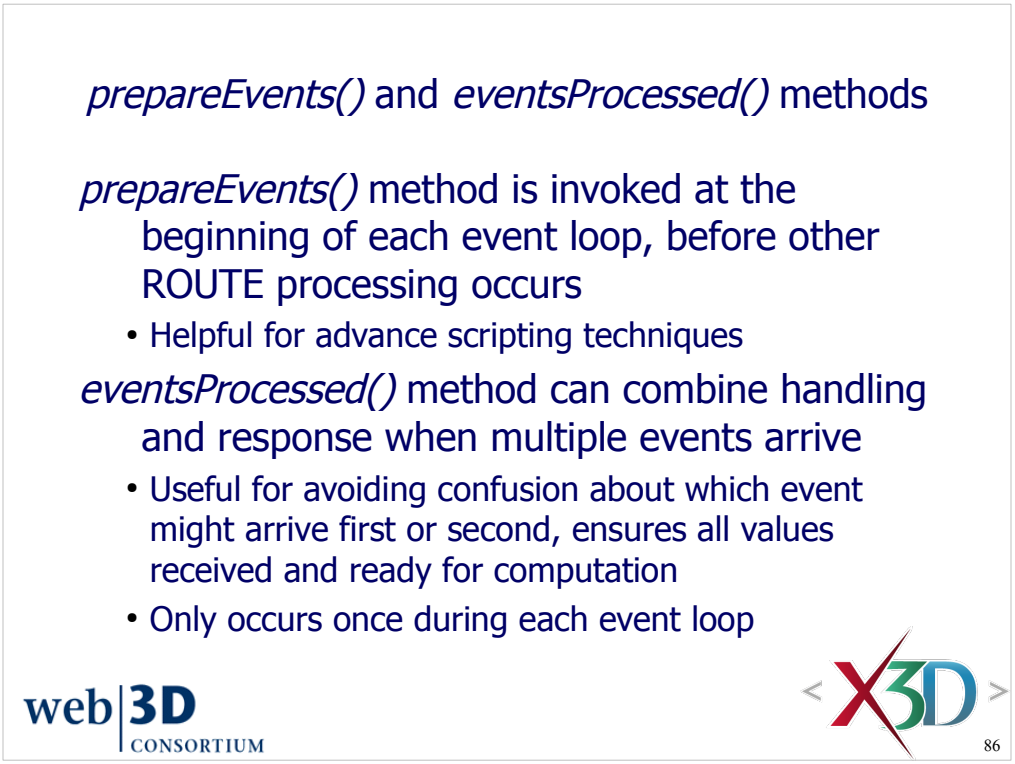
Illustration figures 4 and 5 excerpted from

- Brutzman, Don, “The Virtual Reality Modeling Language and Java,” *Communications of the ACM*, vol. 41 no. 6, June 1998, pp. 57-64.
- http://www.web3d.org/x3d/content/examples/Basic/ScriptConformance/VrmlJavaPaper.pdf

These figures illustrate direct event-based ROUTE control and indirect field-based control, respectively. The examples have been updated from VRML to X3D and retain the same functional structure.

Two sets of examples (using ECMAScript and Java) are provided for each approach.

TODO: proper testing of the revised Java examples using the X3D Scene Access Interface (SAI) interfaces.



Note that these methods support advanced techniques and are seldom used.

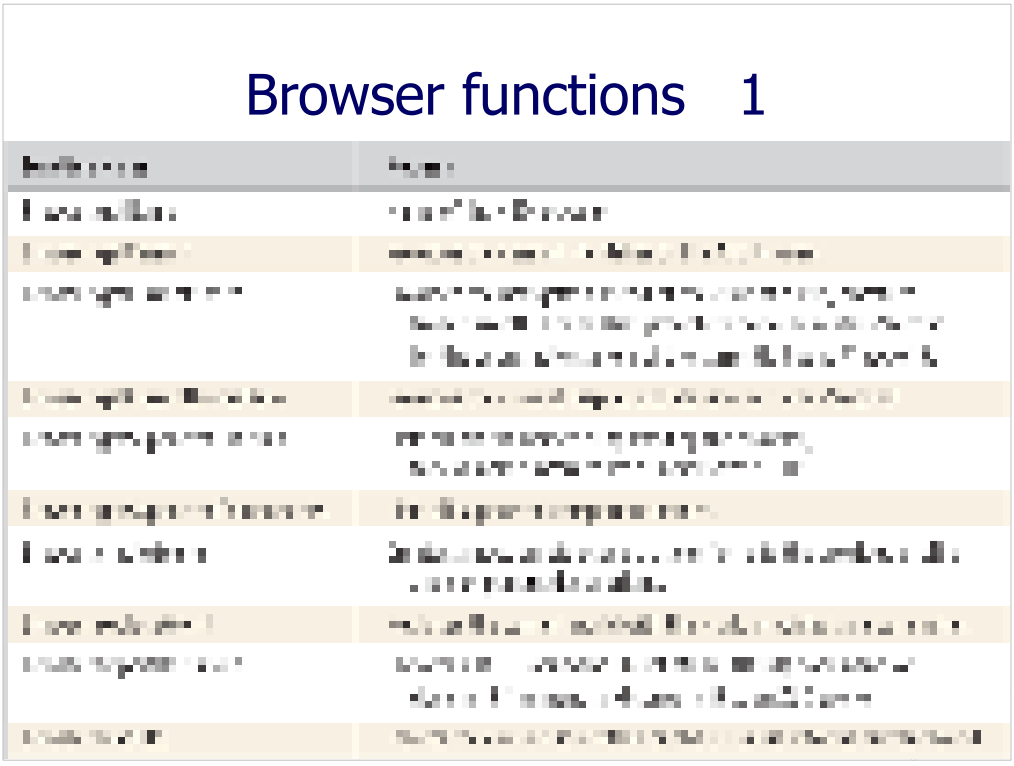


Figure 9.21, page 271, *X3D for Web Authors*.

These functions are provided as part of the Browser class in the EcmaScript bindings for X3D. Authors can use them inside their ecmascript methods.

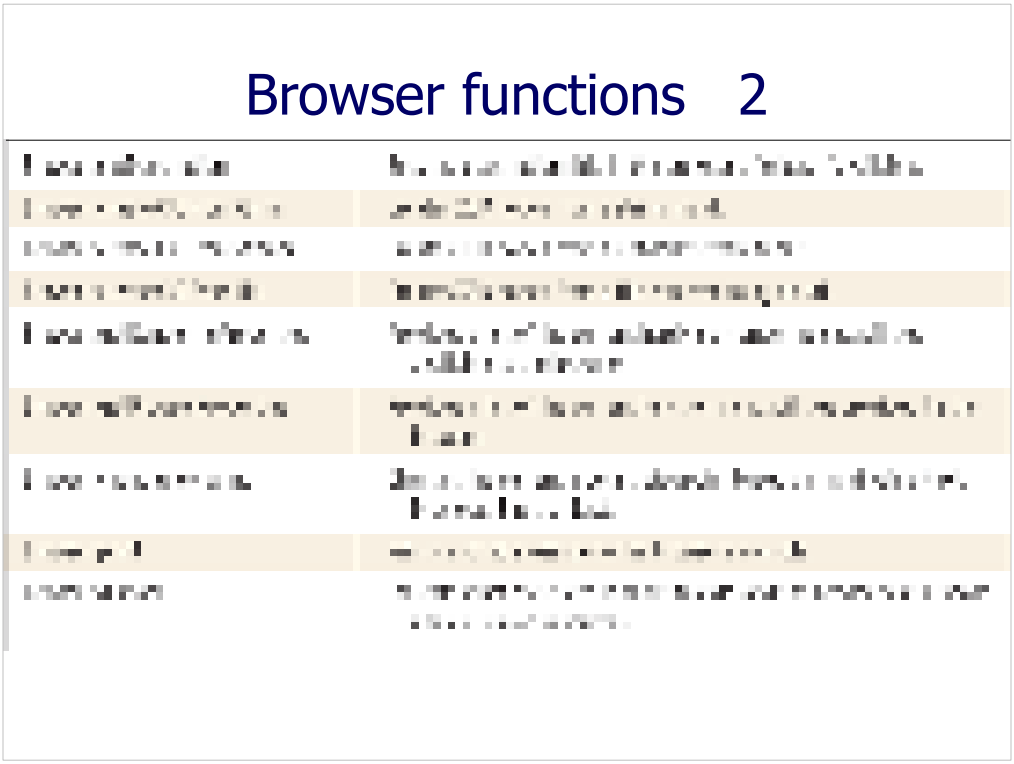
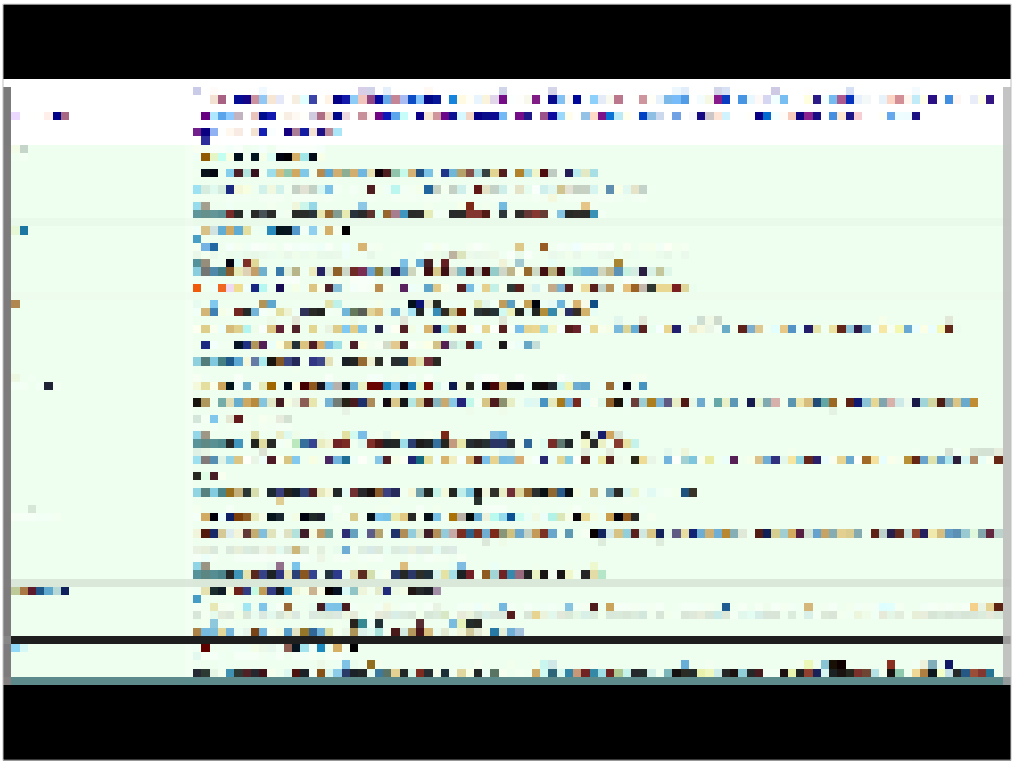
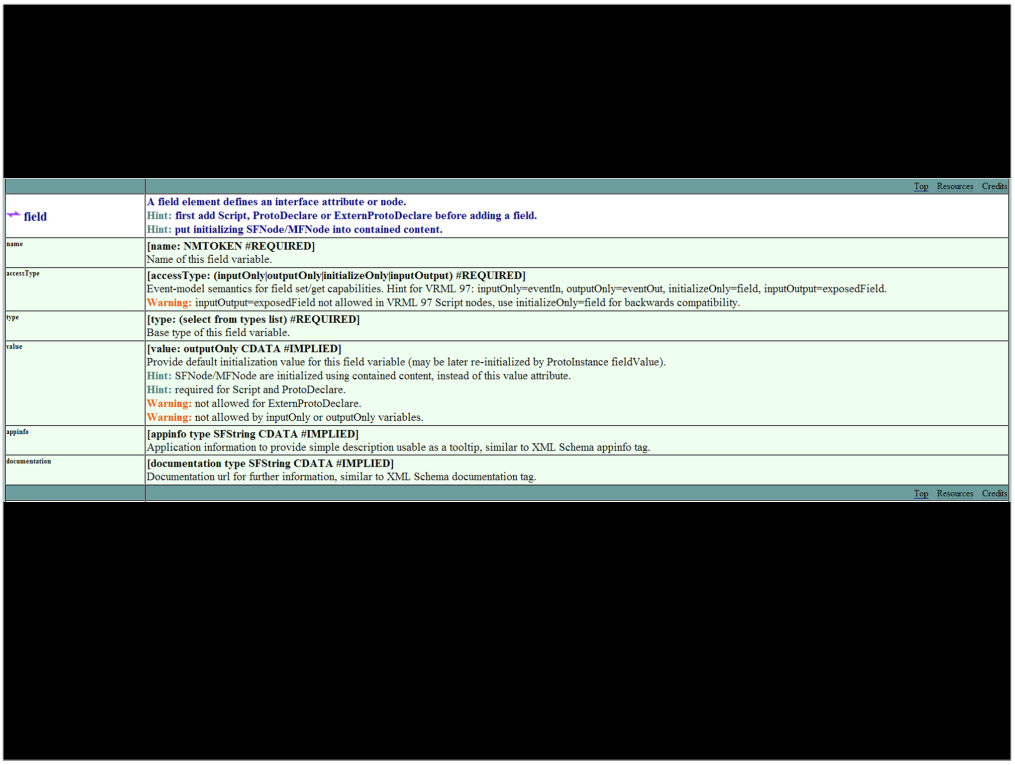


Figure 9.21, page 271, *X3D for Web Authors*.

These functions are provided as part of the Browser class in the EcmaScript bindings for X3D. Authors can use them inside their ecmascript methods.



http://www.web3d.org/x3d/content/X3dTooltips.html#Script



	Top Resources Credits
field	**A field element defines an interface attribute or node.** **Hint: first add Script, ProtoDeclare or ExternProtoDeclare before adding a field.** **Hint: put initializing SFNode/MFNode into contained content.**
name	**[name: NMTOKEN #REQUIRED]** Name of this field variable.
accessType	**[accessType: (inputOnly|outputOnly|initializeOnly|inputOutput) #REQUIRED]** Event-model semantics for field set/get capabilities. Hint for VRML 97: inputOnly=eventIn, outputOnly=eventOut, initializeOnly=field, inputOutput=exposedField. Warning: inputOutput=exposedField not allowed in VRML 97 Script nodes, use initializeOnly=field for backwards compatibility.
type	**[type: (select from types list) #REQUIRED]** Base type of this field variable.
value	**[value: outputOnly CDATA #IMPLIED]** Provide default initialization value for this field variable (may be later re-initialized by ProtoInstance fieldValue). Hint: SFNode/MFNode are initialized using contained content, instead of this value attribute. Hint: required for Script and ProtoDeclare. Warning: not allowed for ExternProtoDeclare. Warning: not allowed by inputOnly or outputOnly variables.
appinfo	**[appinfo type SFString CDATA #IMPLIED]** Application information to provide simple description usable as a tooltip, similar to XML Schema appinfo tag.
documentation	**[documentation type SFString CDATA #IMPLIED]** Documentation url for further information, similar to XML Schema documentation tag.
	Top Resources Credits

http://www.web3d.org/x3d/content/X3dTooltips.html#field

back to Table of Contents

Additional Resources

Additional Resources

AjaX3D is a proposal for use of Asynchronous Javascript for X3D

- Experimental, not part of X3D specification
- http://www.ajax3d.org

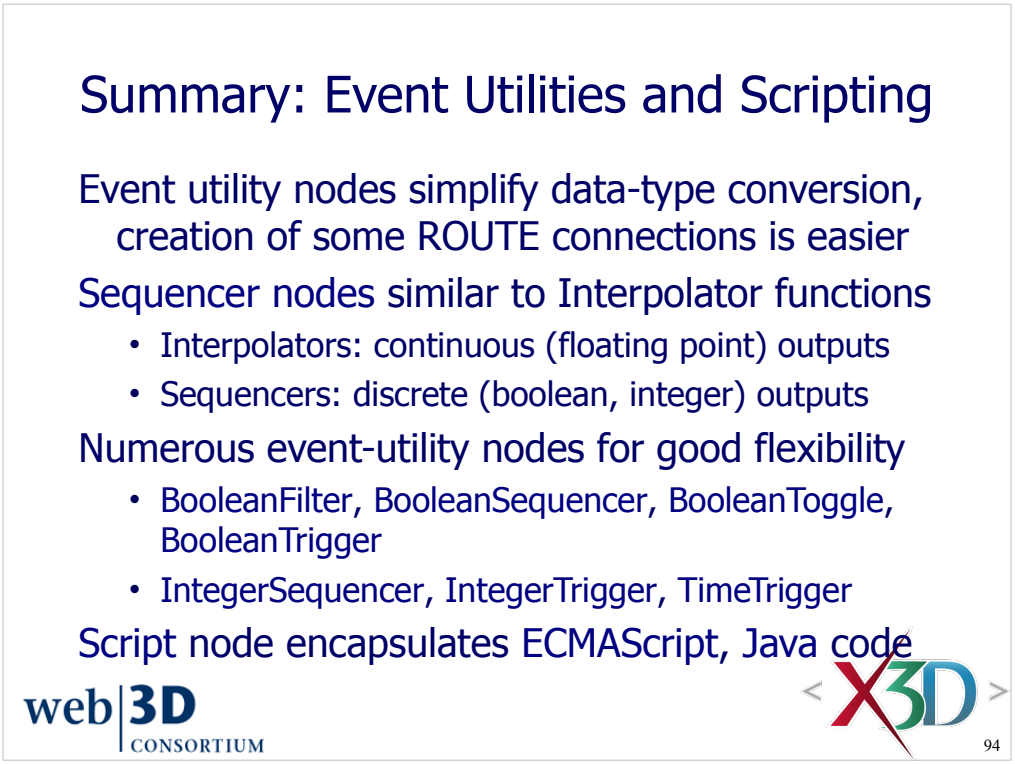
Greg Seidman, Hotpot paper and code

- http://zing.ncsl.nist.gov/~gseidman/vrml/repos
- VRML: Past, Present, and Future
  http://zing.ncsl.nist.gov/~gseidman/class/vrml.html

web3D CONSORTIUM

back to Table of Contents

Chapter Summary



Interpolator nodes are described in Chapter 7, Event Animation and Interpolation. It is especially important to understand the 10-step process for designing and building an animation chain.

Sensor nodes (described in Chapter 8) are also used to produce events that we can ROUTE to other nodes in the scene graph.

The event utilities make it possible (and even easy) to create longer, more sophisticated event chains and behaviors.

Suggested exercises

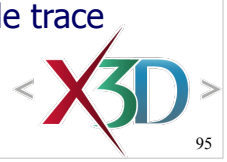
Draw, build event chains with Event Utility nodes

- Combination of boolean and time converters, especially in combination with TimeSensor
- IntegerSequencer, IntegerTrigger with Switch node
- Hey, how about a "Rube Goldberg" animation?! ☺

Build Script node to convert between data types

- Example: StringSensor input for rotation angle, converted to SFRotation about y-axis
- Utilize `initialize()`, `Browser.print()` and `Browser.println()` functions to provide trace statements for debugging

web3D CONSORTIUM



95

http://en.wikipedia.org/wiki/Rube_goldberg

Rube Goldberg Machine Contest

http://en.wikipedia.org/wiki/Rube_Goldberg_Machine_Contest

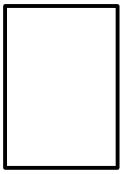
"The 2008 national contest had the task of building a hamburger with a meat patty, two vegetables and two condiments in 20 or more steps."

back to Table of Contents

References

References 1

X3D: Extensible 3D Graphics for Web Authors by Don Brutzman and Leonard Daly, Morgan Kaufmann Publishers, April 2007, 468 pages.



- Chapter 9, Event Utilities and Scripting
- http://x3dGraphics.com
- http://x3dgraphics.com/examples/X3dForWebAuthors

X3D Resources

- http://www.web3d.org/x3d/content/examples/X3dResources.html

web|3D CONSORTIUM

97

References 2

X3D-Edit Authoring Tool

- https://savage.nps.edu/X3D-Edit

X3D Scene Authoring Hints

- http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html

X3D Graphics Specification

- http://www.web3d.org/x3d/specifications
- Also available as help pages within X3D-Edit

web|3D CONSORTIUM

98

References 3

VRML 2.0 Sourcebook by Andrea L. Ames, David R. Nadeau, and John L. Moreland, John Wiley & Sons, 1996.

- http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm
- http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook
- Chapter 30 - Scripts

ECMAScript ECMA-262 Specification

- http://www.ecma-international.org/publications/standards/Ecma-262.htm
- http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf
- http://www.web3d.org/specifications/Ecma-262.pdf

web|3D CONSORTIUM

References 4

Late Night VRML 2.0 with Java

- by Bernie Roehl, Justin Couch, Cindy Reed-Ballreich, Tim Rohaly and Geoff Brown
- Ziff-Davis Press (Macmillan Publishers), 1997
- http://www.ece.uwaterloo.ca/~broehl/vrml/lnvj

web3D CONSORTIUM

<X3D>

100

This book is a tremendous resource with numerous examples that deserve to be upgraded from experimental VRML97 code into X3D.

Contact

Don Brutzman

brutzman@nps.edu

http://faculty.nps.edu/brutzman

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA
1.831.656.2149 voice

web3D CONSORTIUM

101

CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- http://cgems.inesc.pt

X3D for Web Authors recognized by CGEMS! ☺

- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics


From the CGEMS home page:

- http://cgems.inesc.pt

Welcome to CGEMS - Computer Graphics Educational Materials Source. The CGEMS site is designed for educators to provide a source of refereed high-quality content as a service to the Computer Graphics community as a whole. Materials herein are freely available and directly prepared for your classroom.

List of all published modules:

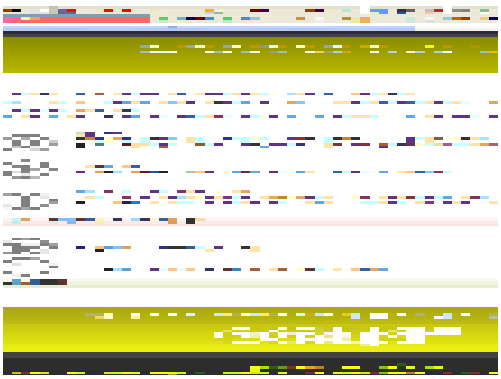
- http://cgems.inesc.pt/authors/ListModules.aspx

CGEMS Editorial Policy:

- http://cgems.inesc.pt/EditorialPolicy.htm

Creative Commons open-source license

http://creativecommons.org/licenses/by-nc-sa/3.0



Open-source license
for X3D-Edit software and X3D example scenes

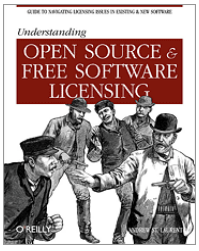
http://www.web3d.org/x3d/content/examples/license.html

License available at

http://www.web3d.org/x3d/content/examples/license.txt

http://www.web3d.org/x3d/content/examples/license.html

Good references on open source:



Andrew M. St. Laurent, *Understanding Open Source and Free Software Licensing*, Oreilly Publishing, Sebastopol California, August 2004. http://oreilly.com/catalog/9780596005818/index.html

Herz, J. C., Mark Lucas, John Scott, *Open Technology Development: Roadmap Plan*, Deputy Under Secretary of Defense for Advanced Systems and Concepts, Washington DC, April 2006. http://handle.dtic.mil/100.2/ADA450769