



Анимация в X3D

- Основные принципы анимации в X3D.
- Каркас событий.
- Генераторы событий.
- Узлы, принимающие события.
- X-ITE for X3D



Рекомендуемая литература

1. А.В. Аксенов. «Интерактивная компьютерная графика». Учебно-методическое пособие. СПб.: ГУАП.2020г. 89с.
2. Аксенов А.В. Каталог примеров по X3D.[Электронный ресурс] URL: <https://aksenov.in/guap/x3dom/> (дата обращения 06.09.2023)
3. Официальная X3DOM документация. Fraunhofer [Электронный ресурс] URL: <https://www.x3dom.org/>(дата обращения 14.09.2023)
4. Архитектура и базовые компоненты X3D. ISO/IEC 19775-1:2013. [Электронный ресурс] URL: <https://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/Architecture.html> (дата обращения 12.09.2023).

Компьютерная анимация

Компьютерная анимация, по сути, является цифровым преемником традиционных методов покадровой анимации. Для создания иллюзии движения изображение выводится на монитор компьютера и неоднократно заменяется новым изображением, похожим на него, но немного опережающим по времени (обычно со скоростью 24, 25 или 30 кадров в секунду).

Эта техника идентична тому, как достигается иллюзия движения на телевидении и в кино. Различия между ключевыми кадрами автоматически вычисляются компьютером в процессе анимации и/или трансформации. Кадры также могут воспроизводиться в реальном времени по мере того, как они представляются конечному пользователю.

Компьютерная анимация с низкой пропускной способностью, передаваемая через Интернет (например, Adobe Flash, X3D), часто использует программное обеспечение на компьютере конечного пользователя для рендеринга в реальном времени, в отличие от потоковой или предварительно загруженной анимации с высокой пропускной способностью.

- Аналогично традиционной, в компьютерной 3D анимации различают :
- анимацию «от позы к позе» (Pose to pose animation).
 - анимацию прямолинейного движения (Straight ahead animation).



Основные принципы анимации

Анимации в X3DOM основаны на анимации ключевых кадров.

Ключевой кадр определяет начальную и конечную точки любого плавного перехода, а путь между ними будет интерполирован.

По умолчанию интерполяция объектов между промежуточными и крайними позами равномерно распределена и отображается в виде прямой линии от одного значения к другому.

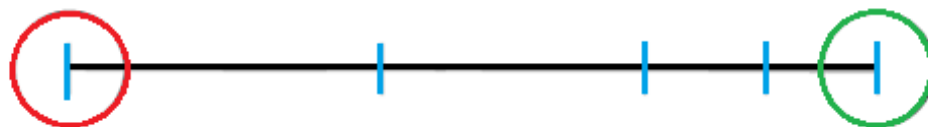
Линейная интерполяция создает довольно монотонное и резкое движение.

Для создания ускорения и замедления движения можно использовать сплайны.

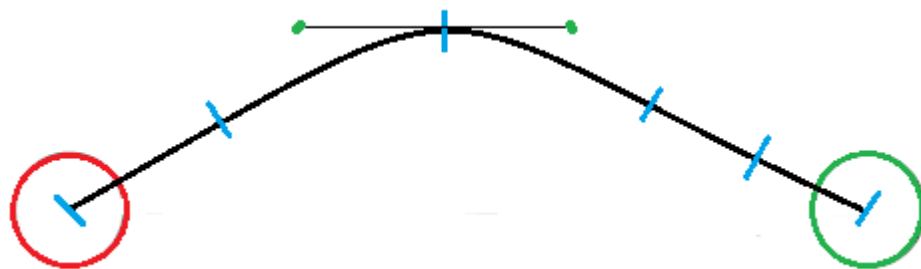
Сплайновая интерполяция создает более выразительное движение.



Линейная интерполяция



Нелинейная интерполяция



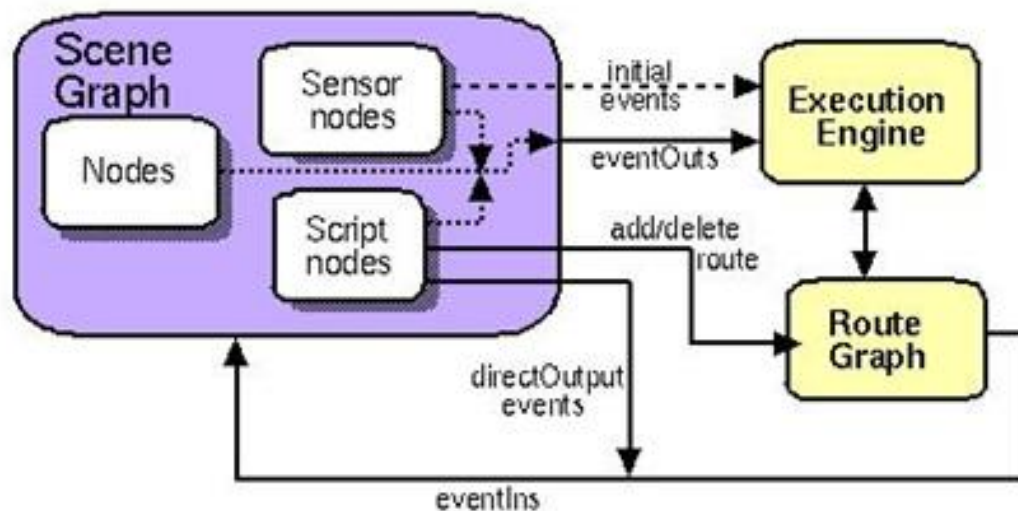
Сплайновая интерполяция



Каркас событий

Анимация X3D-объектов непосредственно связана с изменениями графа сцены. Наличие в стандарте X3D методов, позволяющих объектам X3D-сцен взаимодействовать с пользователем и друг с другом с помощью событий (events), дают возможность создавать динамические сцены.

События, генерируемые через определенные промежутки времени, позволяют создавать динамически изменяющиеся геометрические объекты. Синтаксис узлов включает в числе параметров(полей) каркас событий.



Взаимосвязь событий. Передача события от одного узла к другому, осуществляется при помощи маршрутов объявлением ключевого слова ROUTE.

С их помощью происходит связывание действий, происходящих во времени и\или пространстве, т.е. их маршрутизация от одного узла к другому. Эти связи представлены графом маршрутов и дают возможность контролировать наступление того или иного события, сопровождающегося изменением свойств геометрических узлов.



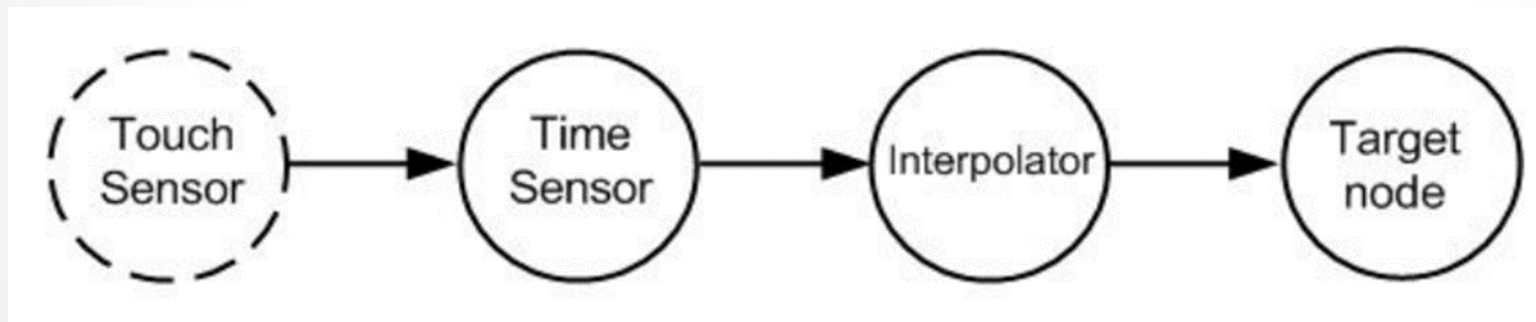
Маршруты ROUTE в X3D

Для того чтобы одни события вызывали друг друга для анимации и интерактивности, необходимо связывать их между собой маршрутами.

Маршрут задается с помощью узла ROUTE.

В атрибутах узла ROUTE указывается:

- fromNode – DEF-имя узла, являющегося источником выходного события;
- fromField – выходное событие этого узла;
- toNode – DEF-имя узла, являющегося приёмником входного события;
- toField – выходное событие этого узла.





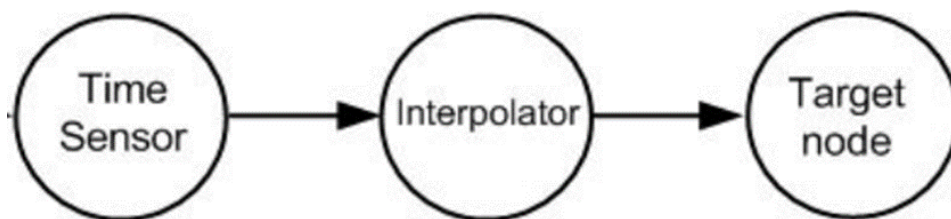
Маршруты ROUTE - Линейная анимация

Для того чтобы одни события вызывали друг друга с целью получения анимации, необходимо связывать их между собой маршрутами.

Маршрут задается с помощью узла ROUTE.

В атрибутах узла ROUTE указывается:

- fromNode – DEF-имя узла, являющегося источником выходного события;
- fromField – выходное событие этого узла;
- toNode – DEF-имя узла, являющегося приёмником входного события;
- toField – выходное событие этого узла.



```
<!--Интерполятор ориентации в пространстве-->
<TimeSensor DEF='Time' cycleInterval='5' loop='true'></TimeSensor>
<OrientationInterpolator DEF='OrientInt' key='0 0.5 1' keyValue='0 0 1 0 0 0 1 -3.14 0 0 1 -6.28'>
</OrientationInterpolator>
<ROUTE fromNode='Time' fromField='fraction_changed'
toNode='OrientInt' toField='set_fraction'>
</ROUTE>
<ROUTE fromNode='OrientInt' fromField='value_changed'
toNode='Sphere' toField='set_rotation'>
</ROUTE>
```



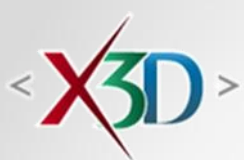

Средства поддержки интерактивности (Генераторы событий - сенсоры)

Примеры применения сенсоров (Sensors) см. по ссылке:

<https://x3dgraphics.com/examples/X3dForWebAuthors/Chapter08UserInteractivity/UserInteractivitySensorNotesIndex.html> (дата обращения 12.09.2023)

В сцены с пользователем, т.е. интерактивность:

- **TimeSensor** – Таймер. Генерирует события в заданном интервале времени.
- **TouchSensor** – сенсор соприкосновений. Отслеживает местоположение и состояние манипулятора и определяет, когда пользователь нажимает на объекты геометрии, содержащиеся в родительской группе узла TouchSensor.
- **PlaneSensor** – сенсор перемещения. Отслеживает и реализует действия пользователя по перемещению объектов геометрии. Перемещение происходит в одной плоскости.
- **SphereSensor** – сенсор сферического вращения. Сенсор определяет нажатие кнопки мыши, когда указатель находится на привязанной к нему геометрии, его можно прикрепить к геометрическим объектам, используя их в одной группе.
- **CylinderSensor** – сенсор цилиндрического (вокруг вертикальной оси) вращения. В отличие от SphereSensor, этот сенсор позволяет вращать объекты только вокруг оси Y.
- **KeySensor** – сенсор, реагирующий на нажатие клавиш клавиатуры, что позволяет использовать эту информацию для дальнейшей обработки.
- **StringSensor** – динамическая строка, выводимая на экран. Действия пользователя, а так же происходящие в сцене события, могут влиять на содержимое этой надписи посредством использования маршрутов.
- **Scripts** – сценарии взаимодействия пользователя с объектами X3D-сцены



TimeSensor – сенсор времени

Таймер. Генерирует события в заданном интервале времени.

TimeSensor : X3DTimeDependentNode, X3DSensorNode {

SFTime [in,out] cycleInterval 1 (0,∞) - интервал цикла (секунды).

SFBool [in,out] enabled - таймер включен (true) или отключен (false)

SFBool [in,out] loop – зациклен таймер или нет (true/false).

SFTime [in,out] pauseTime 0 (-∞,∞) – время, когда таймер будет поставлен на паузу.

SFTime [in,out] resumeTime 0 – время, когда таймер возобновит работу

SFTime [in,out] startTime 0 (-∞,∞) – время включения таймера.

SFTime [in,out] stopTime 0 (-∞,∞) – время выключения таймера.

SFTime [out] cycleTime– это событие генерируется каждый раз, когда таймер достигает интервала цикла, и имеет значение текущего времени.

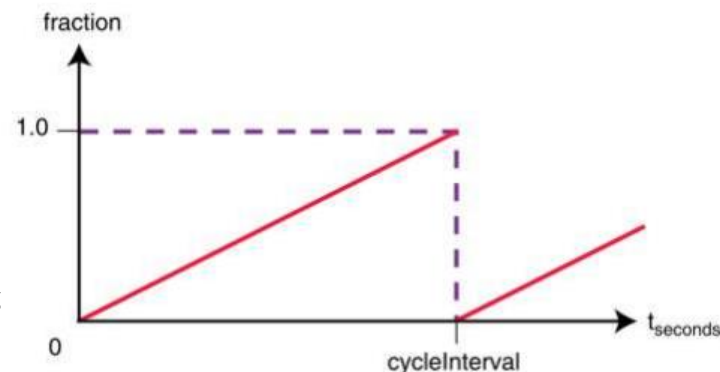
SFTime [out] elapsedTime – общее время работы таймера с момента включения (без учёта пауз)

SFFloat [out] fraction_changed– выходное событие, представляющее собой непрерывный поток сигналов, необходимый для интерполяторов. Событие генерируется постоянно, и так быстро, как возможно. Значение этого события интерполируется во время интервала цикла от 0 до 1 (при достижении таймером значения, кратного интервалу цикла, значение fraction_changed = 1).

SFBool [out] isActive– генерируется, когда таймер начинает работать или останавливается (true – таймер запущен, false – прекратил работу).

SFTime [out] time - генерируется так же, как и fraction_changed, и имеет значение текущего времени.

}

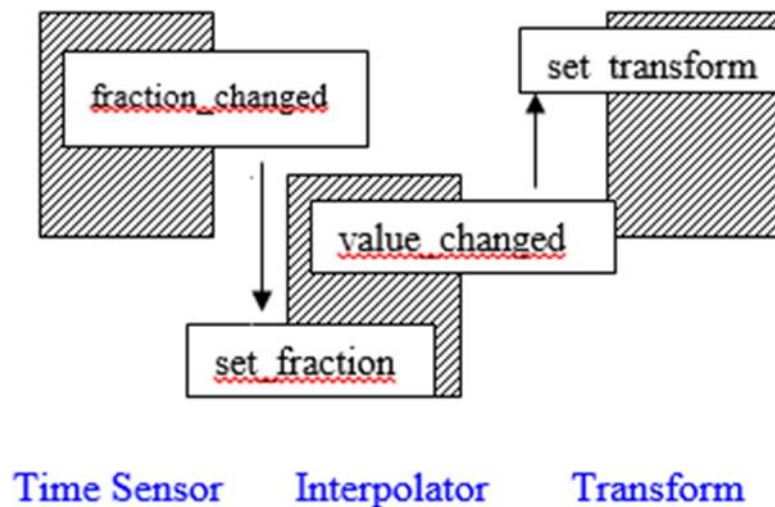


Выходное событие таймера fraction_changed представляет собой пилообразную функцию.



Интерполяторы - узлы, принимающие события

Важными узлами для создания анимированных объектов являются узлы интерполяторы - Interpolator. Основная их особенность состоит в том, что они используются для изменения определенных значений параметров геометрических узлов через определенное время. Временные сигналы узлы Interpolator получают от TimeSensor через определенные интервалы времени и выполняют линейную интерполяцию между значениями, описываемыми keyValues, для каждого значения времени, в долях fraction_changed



Существует множество узлов-интерполяторов, различающихся типом данных интерполируемой величины (тип данных поля keyValue и выходного события value_changed).

Описание общих полей:

- **key** и **keyValue** – задают функцию линейной интерполяции.

Массив **key** перечисляет в неубывающем порядке границы относительных временных интервалов (в диапазоне [0..1]), которым соответствуют значения интерполируемой величины в массиве **keyValue**.

В момент времени, равный элементу массива **key**, выходное событие **value_changed** принимает значение, равное соответствующему элементу массива **keyValue**. В пределах интервалов, определяемых границами массива **key**, для значений **value_changed** осуществляется линейная интерполяция.

- **set_fraction** – входное событие, принимаемое с таймера, значение которого в течение интервала таймера пробегает от 0 до 1 и сравнивается со значениями из **key**, и при совпадении осуществляется переход к следующему участку интерполяции. Соответственно, длительность полного цикла интерполяции равна интервалу цикла таймера. -

value_changed – выходное событие, значение которого интерполируется между значениями из **keyValue**.



Линейная интерполяция

Модель X3D предполагает, что любая анимация во времени может быть аппроксимирована кусочно-линейной функцией. Это позволяет разработчику вручную задавать настолько детализированную функцию анимации, насколько это необ

Для реализации этой концепции используются узлы-интерполяторы, осуществляющие линейную интерполяцию анимируемой величины согласно функции, заданной таблицей значений. Таблица значений задается разработчиком в design-time. Приведенному рисунку соответствует таблица:

key	0	0.2	0.4	0.6	0.8	1
keyValue	0	5	8	9	4	0

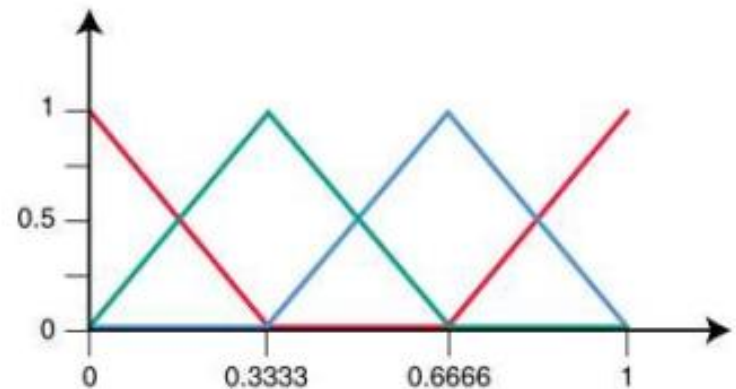
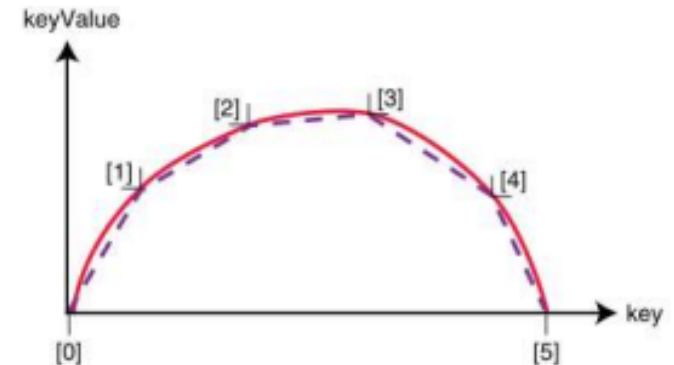
Это реализуется интерполятором скалярной величины следующего вида:

```
<ScalarInterpolator key="0 0.2 0.4 0.6 0.8 1"  
                    keyValue="0 5 8 9 4 0">  
</ScalarInterpolator>
```

В качестве интерполируемого значения не обязательно должна выступать скалярная величина. Им может быть, к примеру, трехкомпонентное значение цвета:

Например:

```
<ColorInterpolator key="0, 0.3333, 0.6666, 1"  
                  keyValue="1 0 0, 0 1 0, 0 0 1, 1 0 0">  
</ColorInterpolator>
```





Узлы-интерполяторы, различающиеся типом данных интерполируемой величины

<https://doc.x3dom.org/tutorials/animationInteraction/onoutputchange/index.html>

- **PositionInterpolator** – интерполятор позиции, выполняет интерполяцию между значениями одиночной координаты (тип SFVec3f),
- **ScalarInterpolator** – интерполятор скалярной величины, выполняет интерполяцию между значениями скалярной величины (одиночного вещественного числа, тип SFFloat).
- **ColorInterpolator** – интерполятор цвета, выполняет интерполяцию между значениями цвета (тройками вещественных чисел 0..1
- **CoordinateInterpolator** – интерполятор координат, выполняет интерполяцию между значениями набора координат в трехмерном пространстве (тип MFVec3f).
- **OrientationInterpolator** – интерполятор ориентации, выполняет интерполяцию между значениями вектора ориентации в пространстве (тип SFRotation).



Пример применения интерполятора изменения цвета

<https://aksenov.in/guap/x3dom/examples/4/colorinterpolator.html>



```
<body>
  <h1>Интерполятор цвета</h1>
  <X3D id="x3d" xmlns="https://www.x3dom.org/x3dom">
    <Scene>
      <TimeSensor DEF='Time' cycleInterval='5' loop='true'></TimeSensor>
      <ColorInterpolator DEF='ColInt' key='0 0.33 0.66 1' keyValue='1 0 0, 0 1 0, 0 0 1, 1 0 0'></ColorInterpolator>
      <Shape>
        <Appearance>
          <Material DEF='Mat'></Material>
        </Appearance>
        <Cylinder></Cylinder>
      </Shape>
      <ROUTE fromNode='Time' fromField='fraction_changed'
        toNode='ColInt' toField='set_fraction'>
      </ROUTE>
      <ROUTE fromNode='ColInt' fromField='value_changed'
        toNode='Mat' toField='set_diffuseColor'>
      </ROUTE>
    </Scene>
  </X3D>
</body>
```



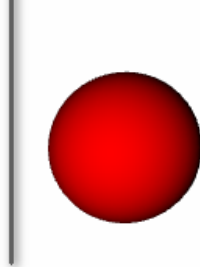
Примеры применения интерполяторов изменения позиции сферы

<https://doc.x3dom.org/tutorials/animationInteraction/onoutputchange/index.html>

```
<body>
<h1> Изменение позиции X3DOM!</h1>
<X3D width='800px' height='400px'>
  <scene>
    <transform DEF="ball" translation='-2 0 0'>
      <shape>
        <appearance>
          <material diffuseColor='1 0 0'></material>
        </appearance>
        <sphere></sphere>
      </shape>
    </transform>

    <timeSensor DEF="time" cycleInterval="4" loop="true"></timeSensor>
    <PositionInterpolator DEF="move" key="0 0.5 1" keyValue="-2 -2.5 0 -2 2.5 0 -2 -2.5 0"></PositionInterpolator>

    <Route fromNode="time" fromField="fraction_changed" toNode="move" toField="set_fraction"></Route>
    <Route fromNode="move" fromField="value_changed" toNode="ball" toField="translation"></Route>
  </scene>
</X3D>
</body>
```





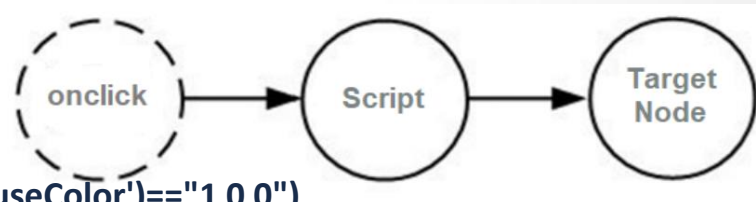
Генераторы событий - Script

Скрипты.

Узел **Script** — позволяет создавать сценарии взаимодействия узлов сцены и событий не предусмотренных стандартным набором сенсоров и интерполяторов X3D

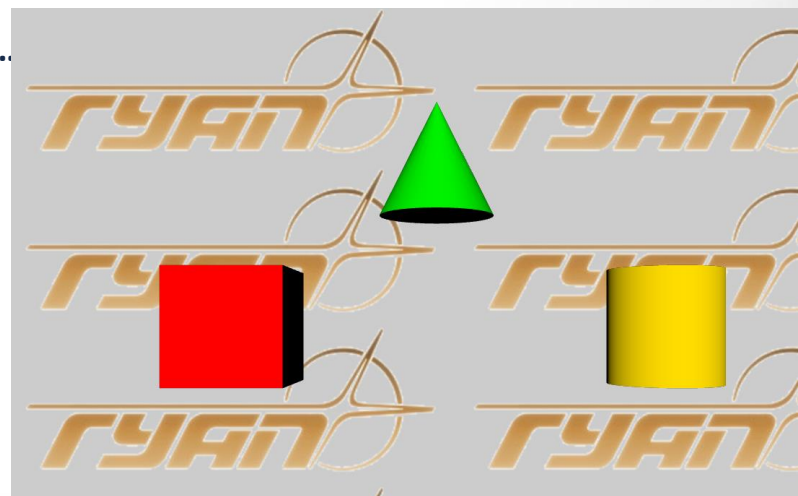
Пример: Прикосновение к объекту

```
<script>
function changeColor()
{
  if(document.getElementById("color").getAttribute('diffuseColor')== "1 0 0")
    document.getElementById("color").setAttribute('diffuseColor', '0 0 1');
  else
    document.getElementById("color").setAttribute('diffuseColor', '1 0 0');
}
</script>
```



...

```
<shape onclick="changeColor();">
  <appearance>
    <material id="color" diffuseColor='1 0 0'></material>
  </appearance>
  <box></box>
</shape>
```



Интерактивные 3D-преобразования

- **Plane Sensor** – датчик перемещения.

```
<group>  
  <planeSensor autoOffset='true'  
    axisRotation='1 0 0 -1.57'  
    minPosition='-6 0'  
    maxPosition='6 0'  
    onoutputchange='processTranslationGizmoEvent(event)';>  
</planeSensor>
```

```
    <transform  
id='translationHandleTransform'>
```

...

```
</transform>
```

```
</group>
```

- **Cylinder Sensor** – датчик цилиндрического (вокруг вертикальной оси) вращения.

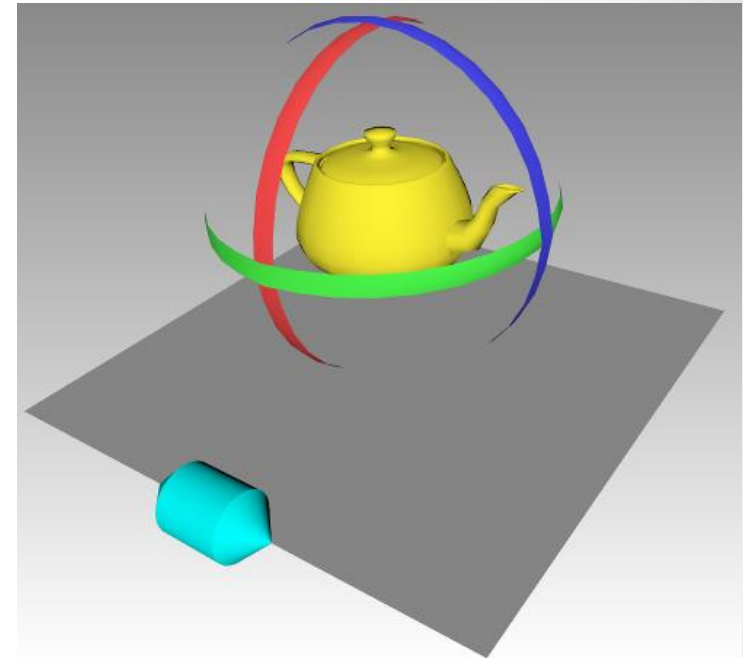
```
<group>  
  <cylinderSensor autoOffset='false' axisRotation='0 0  
1 -1.57'  
  onoutputchange='processRotationGizmoEvent(event);'>  
</cylinderSensor>
```

```
  <transform>
```

...

```
  </transform>
```

```
</group>
```



<https://doc.x3dom.org/tutorials/animationInteraction/transformations/example.html>



TouchSensor – сенсор прикосновения

Touch Sensor – датчик соприкосновений. Узел TouchSensor отслеживает местоположение и состояние устройства ввода(манипулятор) и определяет, когда пользователь указывает на геометрию (нажимает на геометрические объекты), содержащуюся в родительской группе узла TouchSensor.

Этот абстрактный тип узла является базовым типом для всех датчиков указывающих устройств сенсорного типа. TouchSensor : X3DTouchSensorNode {

SFString [in,out] description " - задает текстовое описание узла TouchSensor.

SFBool [in,out] enabled TRUE\FALSE – включает или отключает узел TouchSensor. Если узел TouchSensor отключен, он не отслеживает ввод пользователя и не отправляет события.

SFNode [in,out] metadata NULL [X3DMetadataObject]

SFBool [out] isActive - при нажатии на объекты это событие = TRUE.

SFBool [out] isOver - это событие = TRUE, когда указатель находится над геометрическими объектами, к которым прикреплен датчик. В противном случае, isOver генерирует событие FALSE.

SFTime [out] touchTime - время нажатия на объекты, к которым прикреплен датчик (генерируется при отпускании кнопки мыши).

};

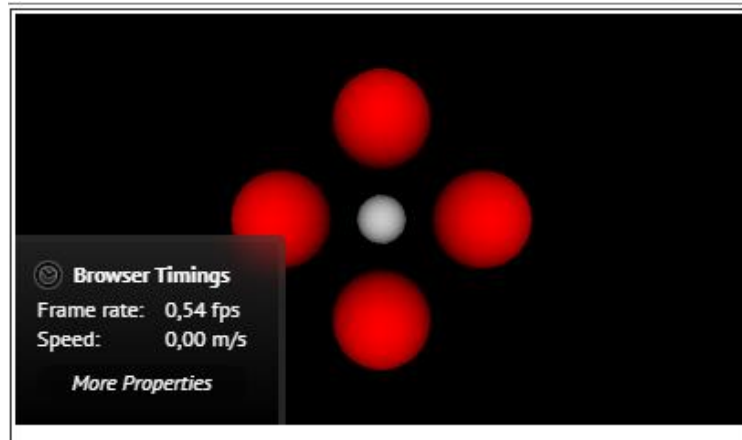
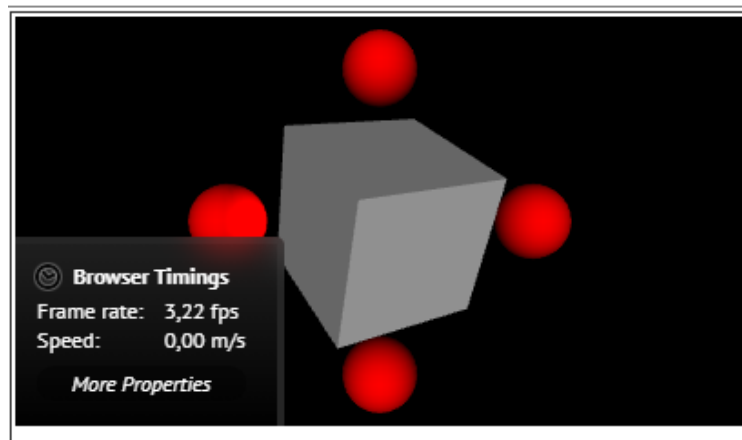
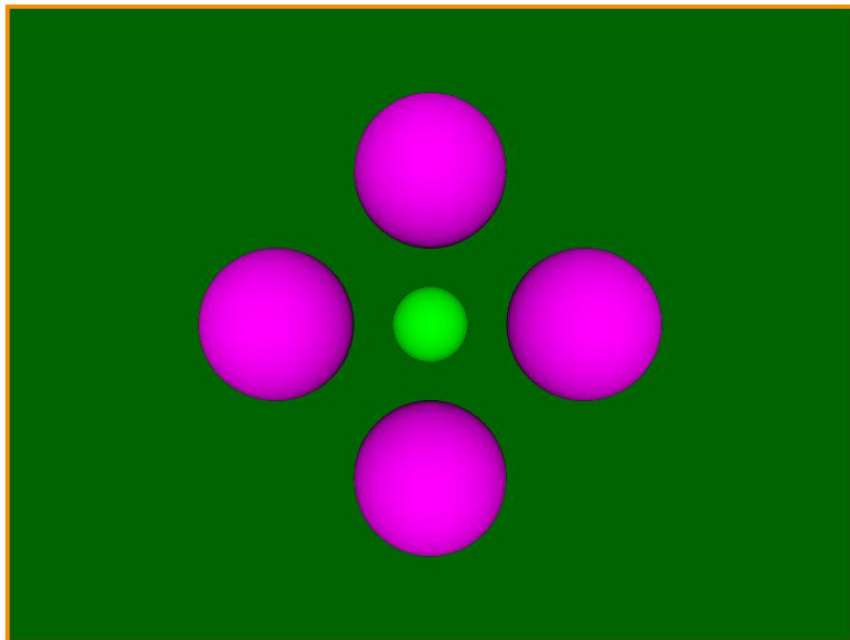
Примеры применения Touch Sensor см. в описании стандарта X3D по ссылке:

<https://www.web3d.org/x3d/content/examples/ConformanceNist/Sensors/TouchSensor/index.html> (дата обращения 12.09.2023)



TouchSensor – сенсор прикосновения

ИКГ Сенсор прикосновения



Примеры применения Touch Sensor см. в описании стандарта X3D по ссылке:

<https://www.web3d.org/x3d/content/examples/ConformanceNist/Sensors/TouchSensor/index.html> (дата обращения 12.09.2023)



X-ITE for X3D

X_ITE — это новая 3D-библиотека JavaScript, полностью написанная на JavaScript и использующая WebGL для 3D-рендеринга.

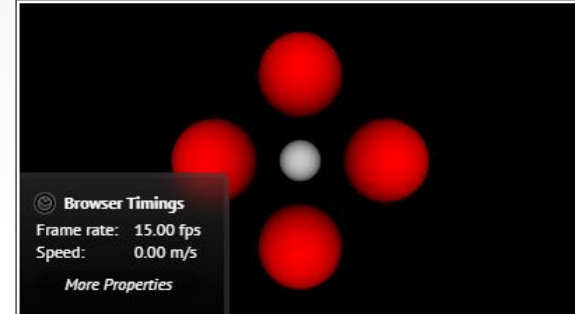
Разработчики X3D-приложений могут публиковать исходный код X3D и VRML в Интернете на странице HTML5 с помощью X_ITE, который работает с веб-браузерами без предварительной установки подключаемого модуля. Это дает возможность разработчикам X3D-сцен отображать контент в 3D, используя технологию трехмерной графики WebGL для отображения контента X3D в нескольких разных браузерах в разных операционных системах.

X3D — это сертифицированный ISO бесплатный формат файлов с открытыми стандартами и архитектура времени выполнения для представления и передачи 3D-сцен и объектов, указанных Консорциумом Web 3D. X_ITE поддерживает пользовательские шейдеры, плоскости отсечения, отображение отражений, узлы сценариев, возможности прототипирования и программирование, управляемое событиями, чтобы предоставить повышенный уровень качества виртуальных эффектов и инструментов, готовых для Интернета.

Примечание разработчика: X_ITE работает с Google Chrome, Chromium, Firefox, Safari, Opera и Microsoft Edge.

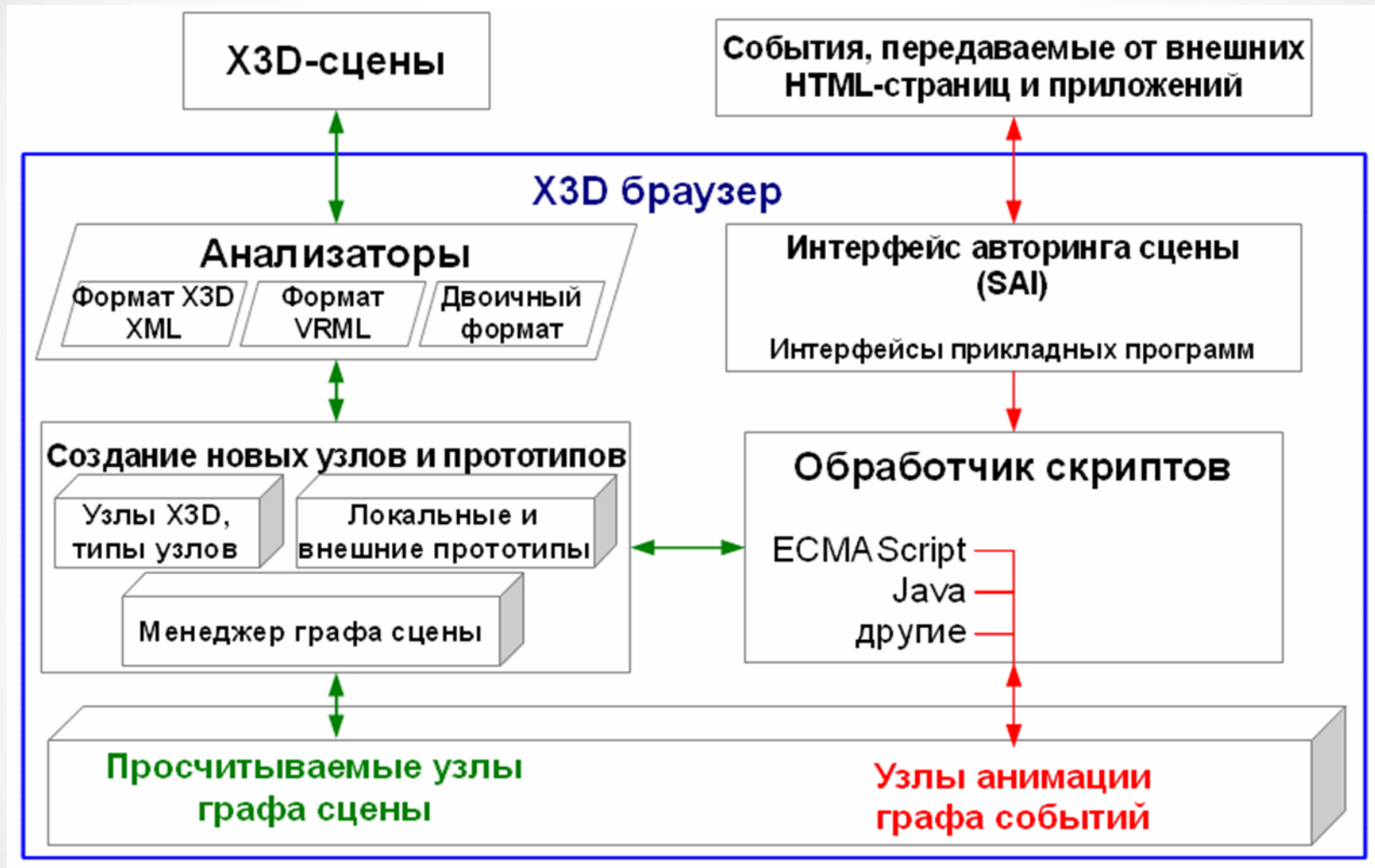
X_ITE использует новейший стандарт JavaScript, но также совместим со старыми браузерами и доступен как на настольном компьютере, так и на современном смартфоне.

1. X-ITE - player x3d. [Электронный ресурс] URL: https://create3000.github.io/x_ite/ (дата обращения 14.09.2023).
2. Пример работы TouchSensor. [Электронный ресурс] URL: <https://create3000.github.io/media/examples/Followers/ColorChaser/example.html> (дата обращения 14.09.2023).
3. X3D Example Archives: Conformance Nist, Sensors, Touch Sensor. [Электронный ресурс] URL: <https://www.web3d.org/x3d/content/examples/ConformanceNist/Sensors/TouchSensor/index.html>





Архитектура программного обеспечения для X3D





Архитектура программного обеспечения для X3D

HTML-страницы и внешние приложения используются для встраивания плагина X3D, который читается браузером и позволяет пользователям взаимодействовать с 3D-сценой на Web-странице.

Анализаторы (Parsers) используются для чтения файлов различных форматов кодировки: X3D, XML, классический VRML или двоичный код.

Затем создаются узлы (Nodes) и отправляются менеджеру (конструктору) графа сцены, который следит за определением геометрических объектов, внешним видом, расположением и ориентацией. Менеджер графа сцены неоднократно обходит узлы, рассчитывая кадры изображения. Этот процесс быстро перерисовывает точно рассчитанные изображения на основе перспективы по мере того как меняется точка зрения пользователя и объекты.

Граф событий также отслеживает все узлы анимации, которые вычисляют изменение событий и передает значение в граф сцены. События, полученные графом сцены, могут изменять положение или свойства геометрических узлов. Дальнейшее расширение узлов анимации реализовано скриптами, которые могут отправлять и получать события, а также создавать (или удалить) геометрические объекты в сцене. Скрипты включают в себя программный код, как правило, на JavaScript или Java.