

Programando em NCL Modelo Conceitual



Copyright © 2012 TeleMídia



1

Entidades Básica

o quê? objetos de mídia

como? propriedades
ou descritores

onde? propriedades
ou regiões

quando? relacionamentos e relações

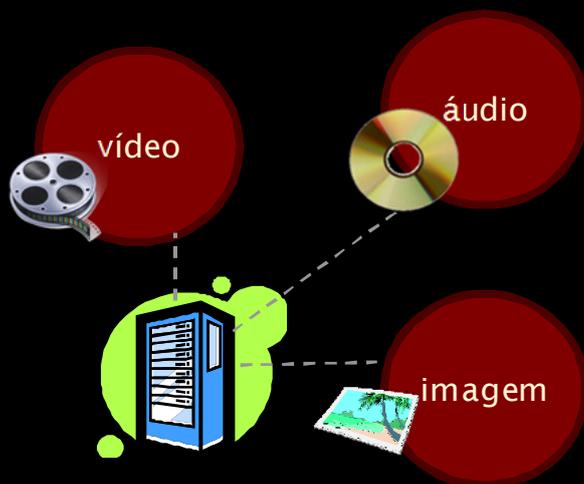


Copyright © 2012 TeleMídia

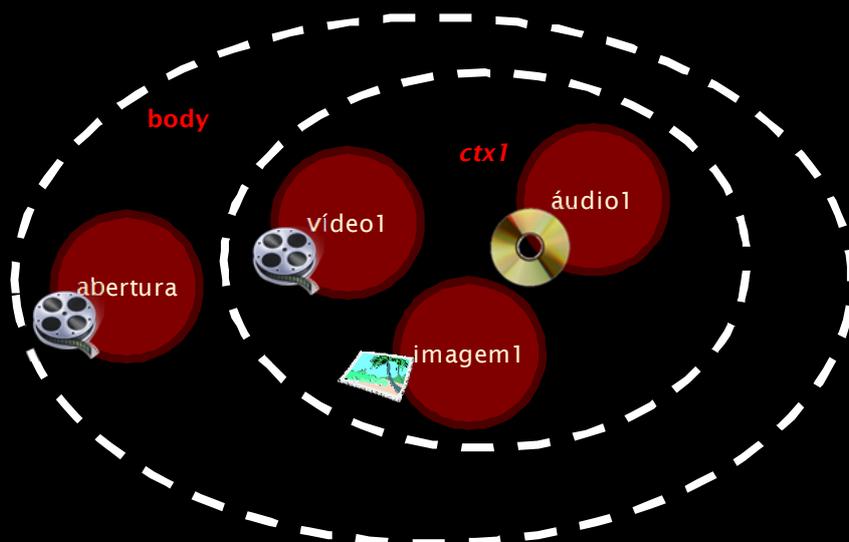


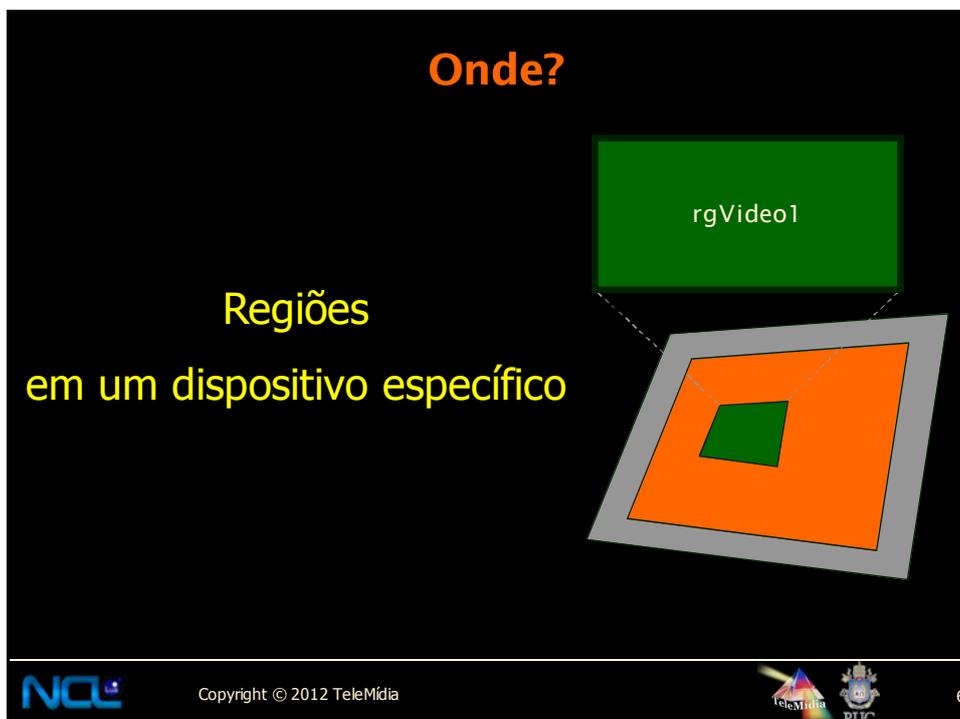
2

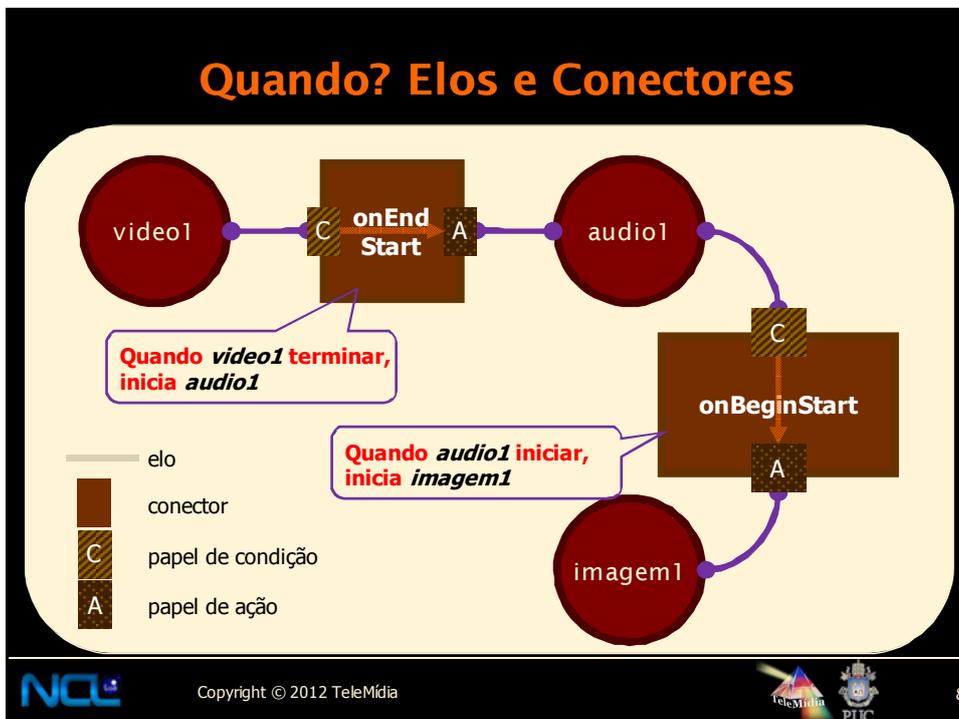
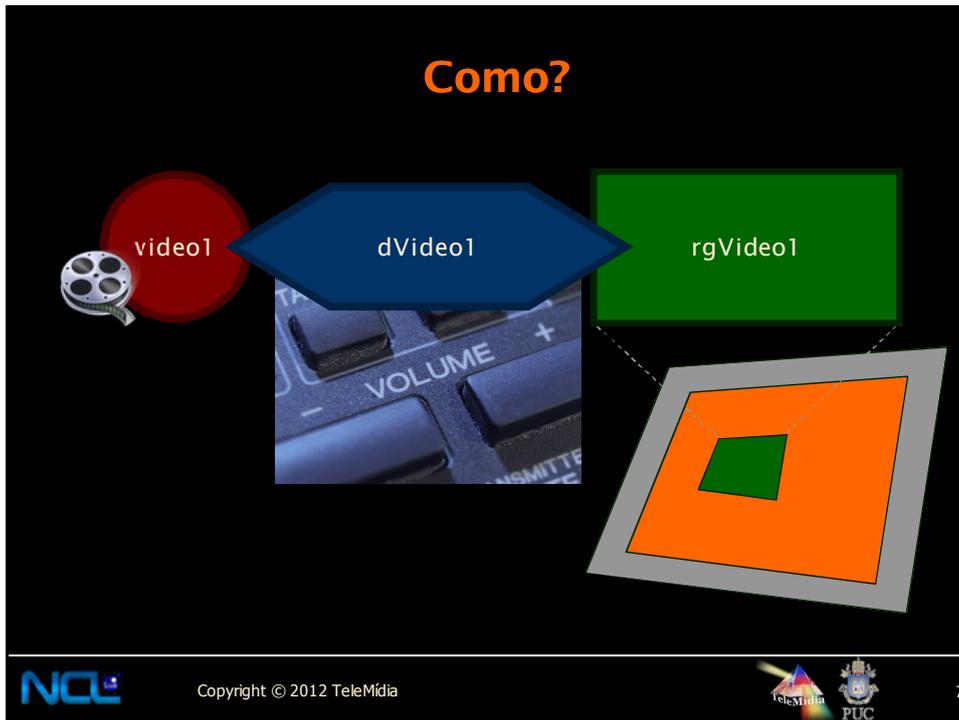
O quê? Objetos de Mídia



Estrutura: contextos







Programando em NCL A Linguagem



Copyright © 2012 TeleMídia



9

Estrutura de um documento NCL – Perfil TV digital

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<ncl id="exemplo00"  
  xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
```

```
<head>
```

cabeçalho do documento

```
</head>
```

```
<body>
```

corpo do documento

```
</body>
```

```
</ncl>
```



Copyright © 2012 TeleMídia



10

Planejando uma aplicação NCL

- visões
 - storyboard
 - estrutural
 - leiaute
 - temporal



Copyright © 2012 TeleMídia



11

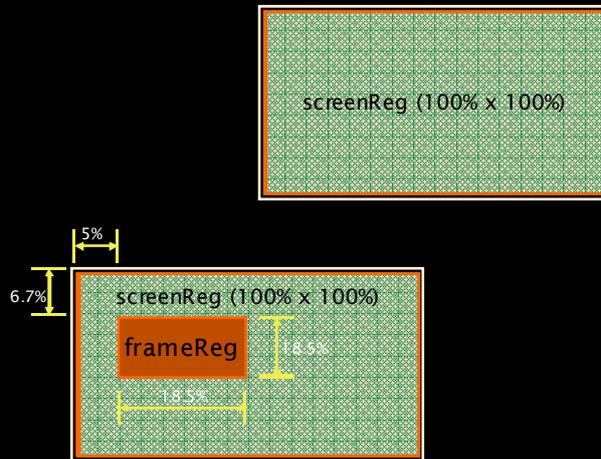
Exemplo 1 – Storyboard



Copyright © 2012 TeleMídia



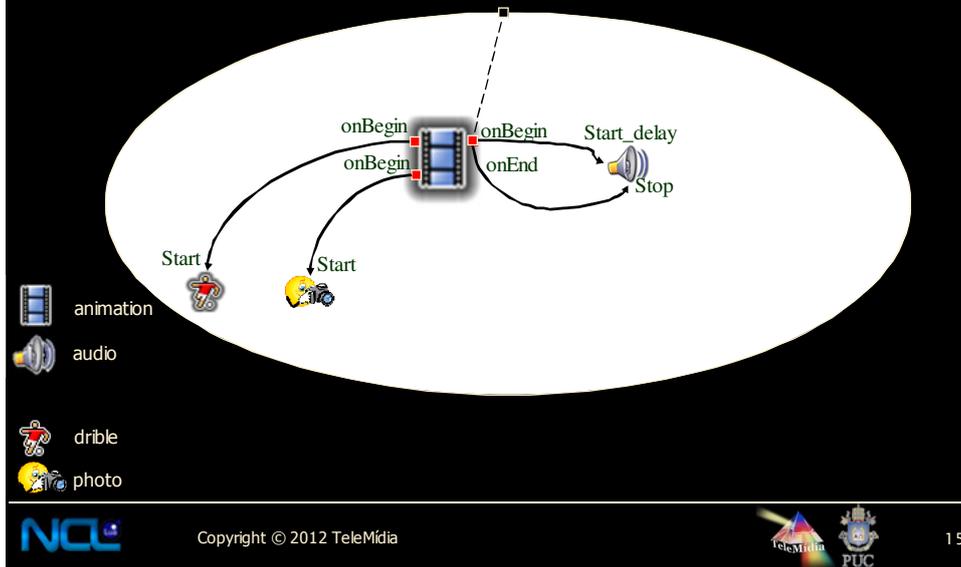
Exemplo 1 - Visão de Leiaute



Exemplo 1 - Visão Temporal



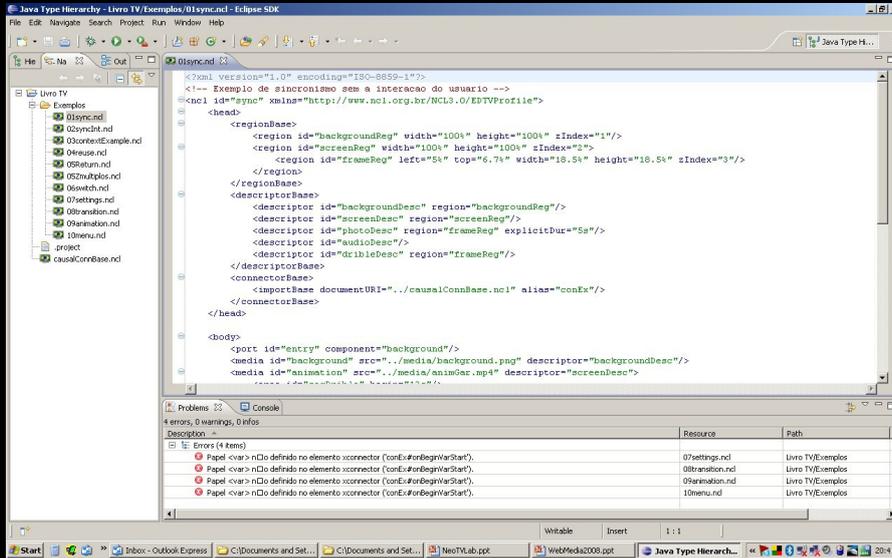
Exemple 1 - Visão Estrutural



NCL Composer

The screenshot shows the NCL Composer interface with several panels. The 'NCL Textual View' on the left displays XML code for a media composition. The 'Structural View' in the center shows a hierarchical tree diagram of the composition. The 'Outline View' on the right lists the elements and their relationships. The 'Properties View' at the bottom left shows the attributes of the selected 'media.m1' element. The 'Layout View' at the bottom center shows a visual representation of the composition's layout. The 'Validator Plugin' on the bottom right shows a list of validation errors. The NCL logo and 'Copyright © 2012 TeleMídia' are at the bottom left, and 'TeleMídia PUC' and the number '15' are at the bottom right.

NCL Eclipse



Copyright © 2012 TeleMídia



17

Set-top Box Virtual Ginga-NCL

Quick hints:

- From your host operating system, use SSH to open a text console. The STB's IP address is shown below.
- User is "root"; Initial password is "telemedia"
- Upload your NCL applications and media to the /misc/ncl30 directory, via SFTP or SCP
- Use the /misc/launchersh script to run your NCL application
Example: /misc/launcher.sh /misc/ncl30/sample03/sample03.ncl
- Use the following keymap:
- Have fun!

IP:192.168.127.129

TV Interativa se faz com Gingal



Copyright © 2012 TeleMídia



18

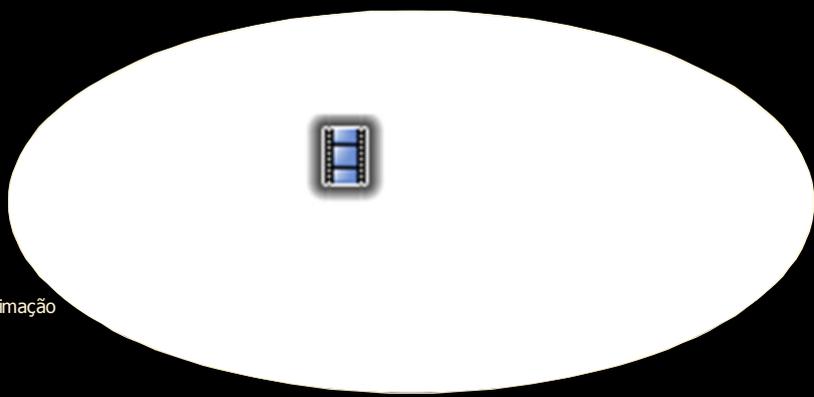


Construindo uma aplicação NCL

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
  
<ncl id="exemplo00" xmlns="http://www.ncl.org.br/NCL3.0/  
EDTVProfile">  
  
  <head>  
    cabeçalho do documento 1  
  </head>  
  
  <body>  
    corpo do documento 2  
  </body>  
</ncl>
```

NCL   20

Exemplo 1 – Visão Estrutural



animação



Copyright © 2012 TeleMídia



21

Exemplo 1

```
<body>
```

```
<media id="animation" src="../../media/animGar.mp4" >
```



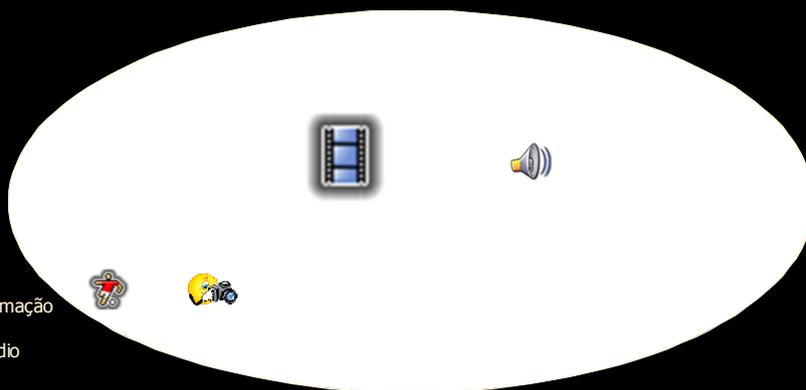
Copyright © 2012 TeleMídia



22

Esquema	Parte específica do esquema	Uso
file:	///file_path/#fragment_identifier	Para arquivos locais
http:	//server_identifier/file_path/#fragment_identifier	Para arquivos remotos buscados pelo canal de interatividade usando o protocolo http
https:	//server_identifier/file_path/#fragment_identifier	Para arquivos remotos buscados pelo canal de interatividade usando o protocolo https
rtsp:	//server_identifier/file_path/#fragment_identifier	Para fluxos (<i>streams</i>) obtidos pelo canal de interatividade usando o protocolo rtsp
rtp:	//server_identifier/file_path/#fragment_identifier	Para fluxos (<i>streams</i>) obtidos pelo canal de interatividade usando o protocolo rtp
ncl-mirror:	//media_element_identifier	Para um fluxo de conteúdo idêntico a um que esteja em apresentação por um outro elemento de mídia
sbtvd-ts:	//program_number.component_tag	Para fluxos elementares recebidos pelo fluxo de transporte (TS)

Exemplo 1 – Visão Estrutural



animação

áudio

drible

foto

Exemplo 1

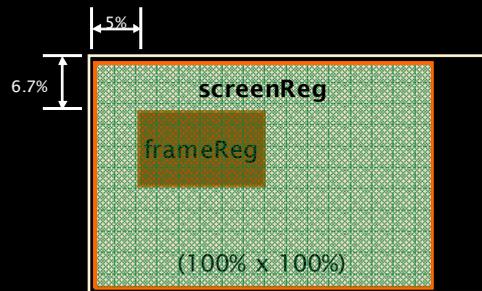
```
<body>  
  
{ <media id="animation" src="../../media/animGar.mp4" >  
  
    <property name="width" value="100%"/>  
    <property name="height" value="100%"/>  
    <property name="zIndex" value="2"/>  
  
    </media>  
    <media id="choro" src="../../media/choro.mp3"/>  
    <media id="drible" src="../../media/drible.mp4">  
        <property name="left" value="5%"/>  
        <property name="top" value="6.7%"/>  
        <property name="width" value="18.5%"/>  
        <property name="height" value="18.5%"/>  
        <property name="zIndex" value="3"/>  
  
    </media>
```

Exemplo 1

```
{ <media id="photo" src="../../media/photo.png">  
    <property name="left" value="5%"/>  
    <property name="top" value="6.7%"/>  
    <property name="width" value="18.5%"/>  
    <property name="height" value="18.5%"/>  
    <property name="zIndex" value="3"/>  
    <property name="explicitDur" value="5s"/>  
  
    </media>
```

Regions

```
<regionBase>  
  <region id="screenReg" height="100%" width="80%" zIndex="2">  
    <region id="frameReg" left="5%" top="6.7%" height="18.5%" width="160"  
                                             zIndex="3"/>  
  </region>  
</regionBase>
```



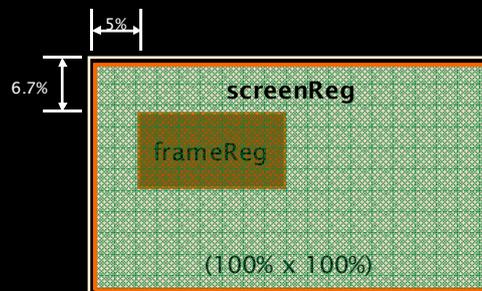
Copyright © 2012 TeleMídia



27

Regions

```
<regionBase>  
  <region id="screenReg" height="100%" width="100%" zIndex="2">  
    <region id="frameReg" left="5%" top="6.7%" height="18.5%" width="160"  
                                             zIndex="3"/>  
  </region>  
</regionBase>
```



Copyright © 2012 TeleMídia



28

Descriptors (how?)

- em que região um objeto será apresentado

```
<descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
```

- por quanto tempo um objeto será apresentado

```
<descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
```

- **Como** um objeto é inicialmente apresentado

- p.ex. volume do áudio (soundLevel),
borda de ums mídia textual (border)

```
<descriptor id="audioDesc" >  
  <descriptorParam name="soundLevel" value="70%" />  
</descriptor>
```

Exemplo 2 - Descriptors

```
<descriptorBase>  
  <descriptor id="screenDesc" region="screenReg"/>  
  <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>  
  <descriptor id="audioDesc"/>  
  <descriptor id="dribleDesc" region="frameReg"/>  
</descriptorBase>
```



```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Exemplo de sincronismo sem a interacao do usuario -->
<ncl id="sync" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>
    <regionBase>
      <region id="screenReg" height="100%" width="100%" zIndex="1">
        <region id="frameReg" left="5%" top="6.7%" height="18.5%" width="18.5%"
          zIndex="3"/>
      </region>
    </regionBase>
    <descriptorBase>
      <descriptor id="screenDesc" region="screenReg"/>
      <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
      <descriptor id="audioDesc"/>
      <descriptor id="dribbleDesc" region="frameReg"/>
    </descriptorBase>
  </head>

```



Copyright © 2012 TeleMídia



31

Exemplo 1

```

<body>
  <media id="animation" src="../../media/animGar.mp4" >
    <property name="width" value="100%"/>
    <property name="height" value="100%"/>
    <property name="zIndex" value="2"/>
  </media>

```



Copyright © 2012 TeleMídia



32

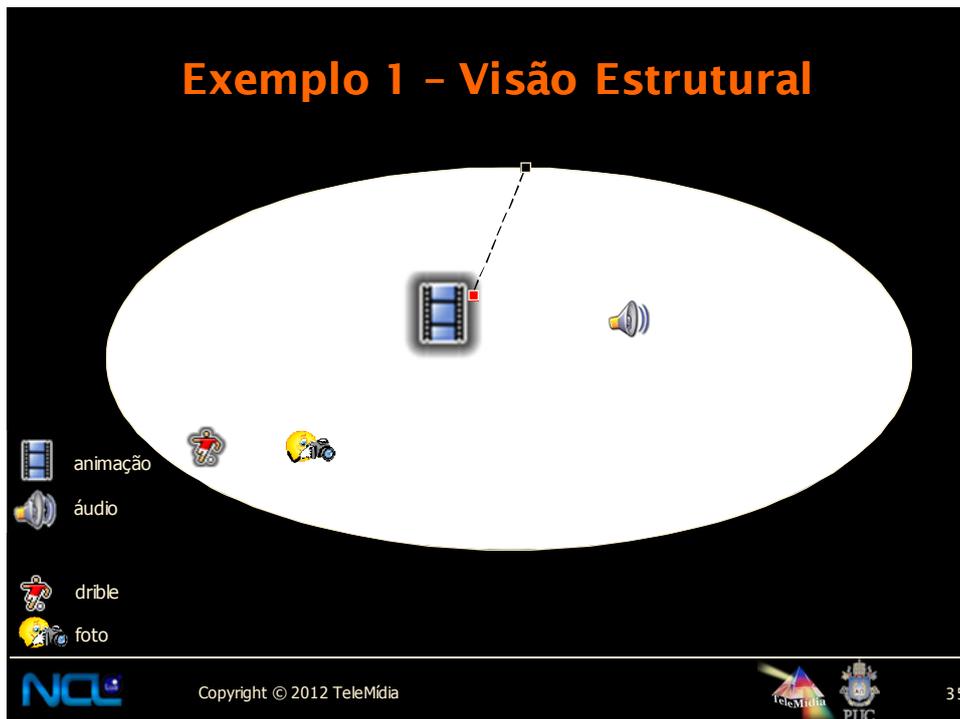
Exemplo 1

```
<body>  
  
{  
  <media id="animation" src="../../media/animGar.mp4" descriptor="screenDesc">  
  
  </media>  
}
```

Exemplo 2

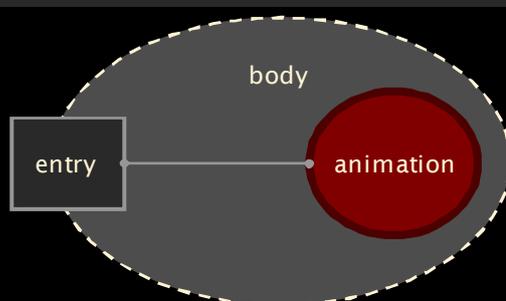
```
<body>  
  
{  
  <media id="animation" src="../../media/animGar.mp4" descriptor="screenDesc">  
  
  </media>  
  <media id="choro" src="../../media/choro.mp3" descriptor="audioDesc"/>  
  <media id="drible" src="../../media/drible.mp4" descriptor="dribleDesc"/>  
  <media id="photo" src="../../media/photo.png" descriptor="photoDesc"/>  
}
```

Exemplo 1 - Visão Estrutural



Exemplo 1 - Contexto *body* e port

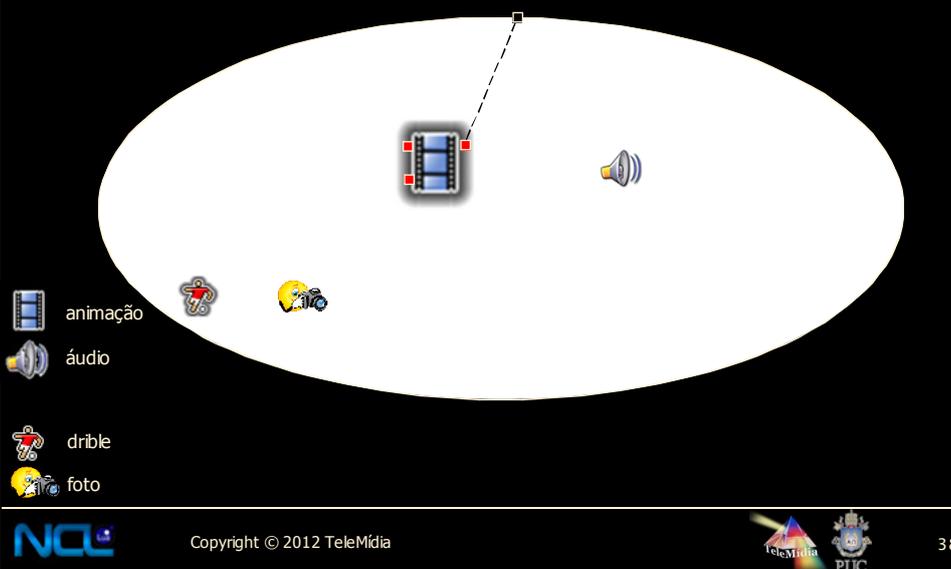
```
<body>  
  <port id="entry" component="animation" />  
  ...  
</body>
```



Exemplo 2

```
<body>
{ <port id="entry" component="animation"/>
  <media id="animation" src="../../media/animGar.mp4" descriptor="screenDesc">
  </media>
  <media id="choro" src="../../media/choro.mp3" descriptor="audioDesc"/>
  <media id="drible" src="../../media/drible.mp4" descriptor="dribleDesc"/>
  <media id="photo" src="../../media/photo.png" descriptor="photoDesc"/>
}
```

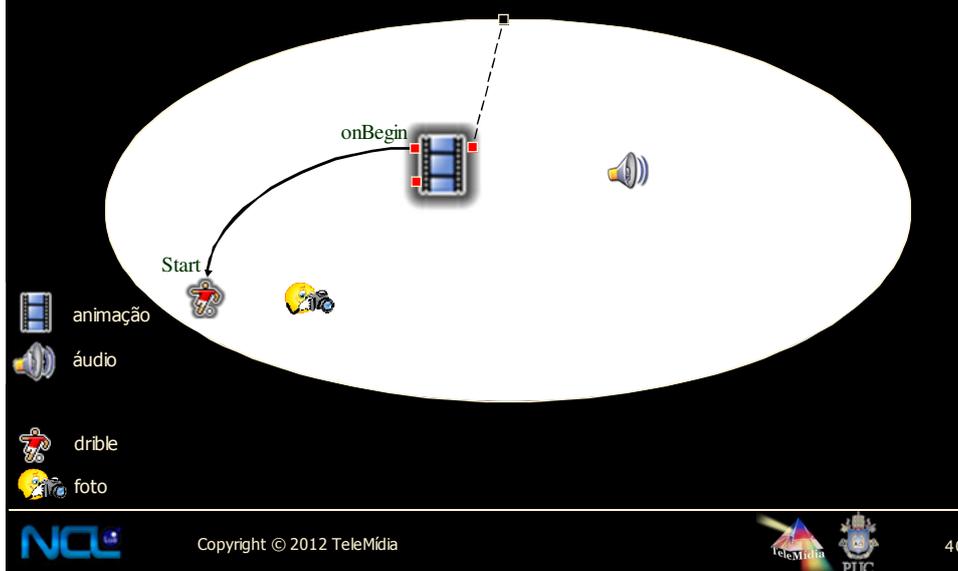
Exemplo 1 - Visão Estrutural



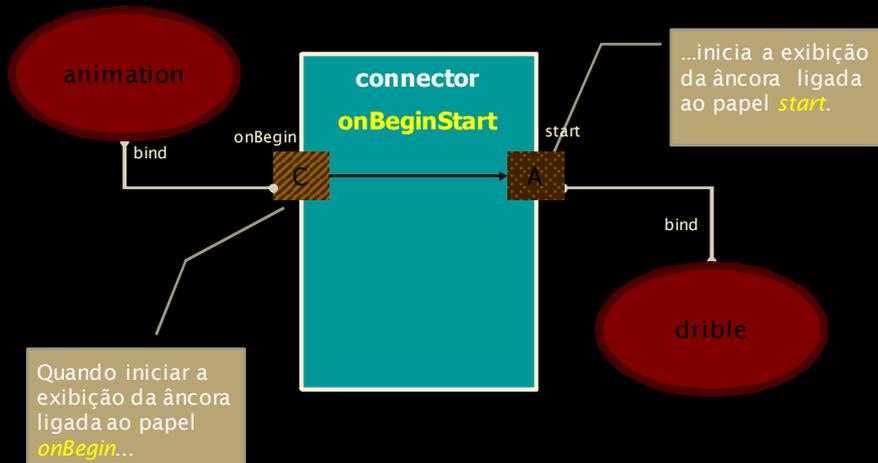
Exemplo 2

```
<body>
{
  <port id="entry" component="animation"/>
  {
    <media id="animation" src="../media/animGar.mp4" descriptor="screenDesc">
      <area id="segDrible" begin="12s"/>
      <area id="segPhoto" begin="41s"/>
    </media>
    <media id="choro" src="../media/choro.mp3" descriptor="audioDesc"/>
    <media id="drible" src="../media/drible.mp4" descriptor="dribleDesc"/>
    <media id="photo" src="../media/photo.png" descriptor="photoDesc"/>
  }
}
```

Exemplo 1 - Visão Estrutural



Connector *onBeginStart*



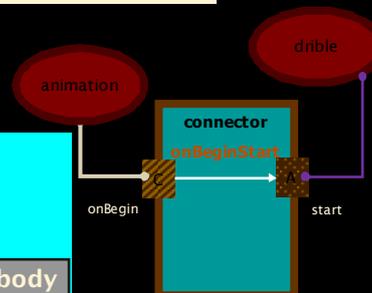
Connector *onBeginStart*

```
<causalConnector id="onBeginStart">
  <simpleCondition role="onBegin"/>
  <simpleAction role="start" max="unbounded" qualifier="seq"/>
</causalConnector>
```

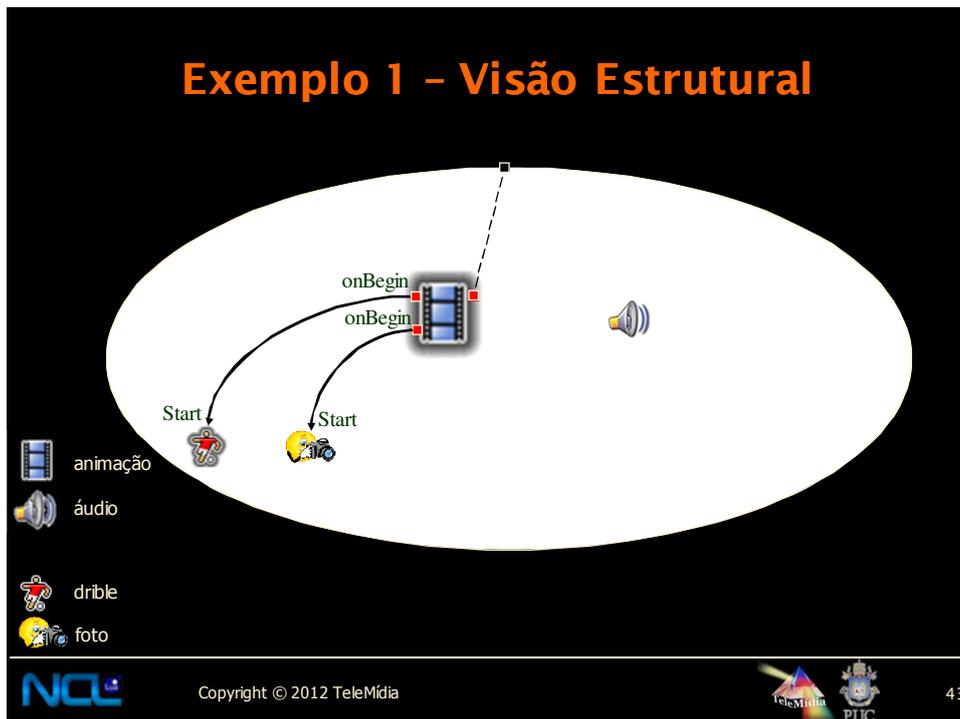
Elo que utiliza o conector *onBeginStart*:
Iniciando o vídeo *drible*

```
<link xconnector="onBeginStart">
  bind role="onBegin" component="animation"
  interface="segDrible/>
  <bind role="start" component="drible" />
</link>
```

na seção *body*



Exemplo 1 – Visão Estrutural



Connector *onBeginStart*

```

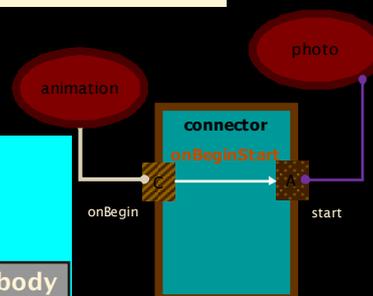
<causalConnector id="onBeginStart">
  <simpleCondition role="onBegin"/>
  <simpleAction role="start" max="unbounded" qualifier="seq" />
</causalConnector>
  
```

Elo que utiliza o conector *onBeginStart*:
Iniciando a imagem da *foto*

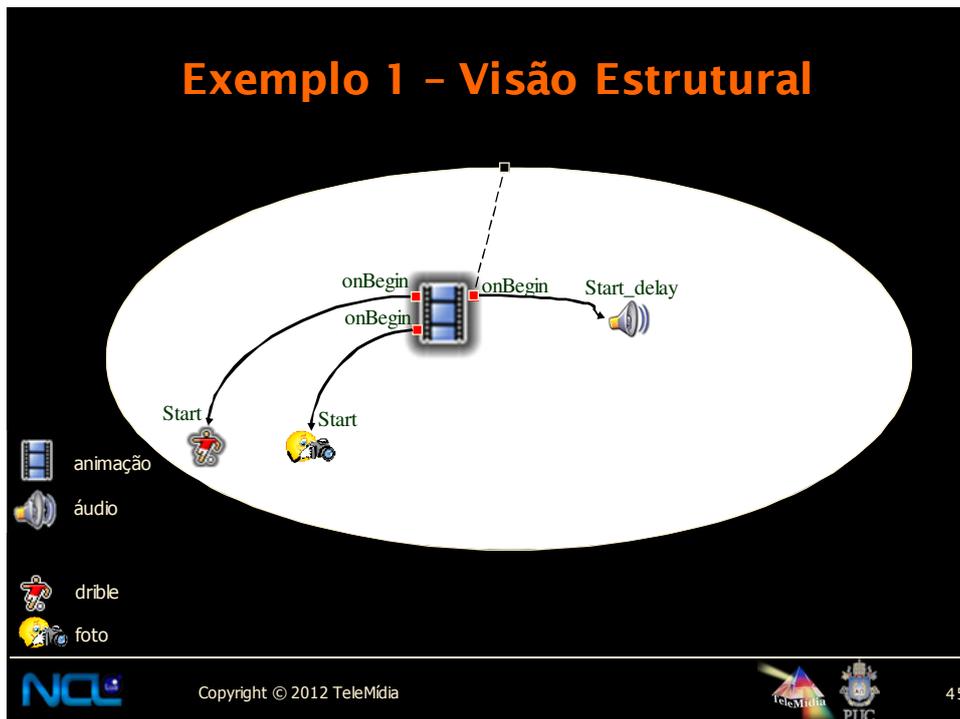
```

<link xconnector="onBeginStart">
  <bind role="onBegin" component="animation"
        interface="segPhoto"/>
  <bind role="start" component="photo" />
</link>
  
```

na seção *body*



Exemplo 1 – Visão Estrutural



Connector *onBeginStart_delay*

```

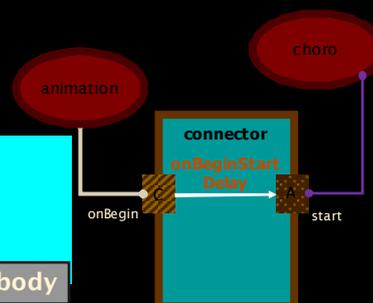
<causalConnector id="onBeginStart_delay">
  <simpleCondition role="onBegin"/>
  <simpleAction role="start" delay="5s" max="unbounded" qualifier="seq"/>
</causalConnector>
  
```

Elo que utiliza o conector *onBeginStart*:
Iniciando o áudio *choro*

```

<link xconnector="onBeginStart_delay">
  <bind role="onBegin" component="animation"/>
  <bind role="start" component="choro" />
</link>
  
```

na seção *body*



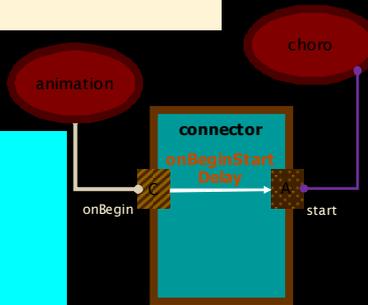
Connector *onBeginStart_delay*

```
<causalConnector id="onBeginStart_delay">
  <connectorParam name="xyz"/>
  <simpleCondition role="onBegin"/>
  <simpleAction role="start" delay="$xyz" max="unbounded" qualifier="seq"/>
</causalConnector>
```

Elo que utiliza o conector *onBeginStart*:
Iniciando o áudio *choro*

```
<link xconnector="onBeginStart_delay">
  <bind role="onBegin" component="animation"/>
  <bind role="start" component="choro">
    <bindParam name="xyz" value="5s"/>
  </bind>
</link>
```

na seção *body*

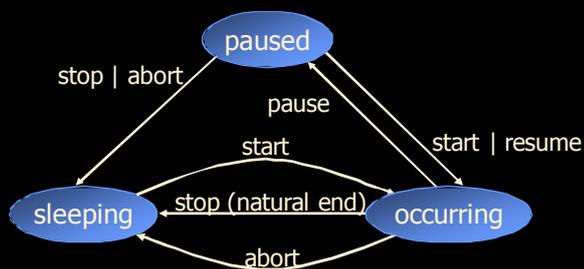


Copyright © 2012 TeleMídia



47

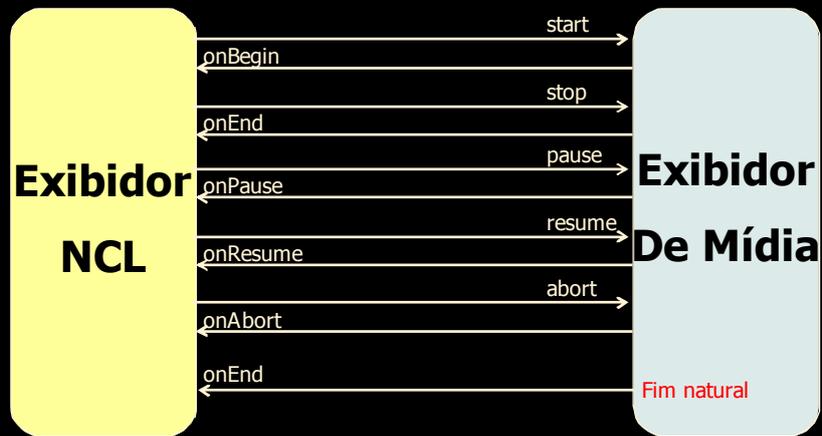
Máquina de Estado de Evento



Copyright © 2012 TeleMídia



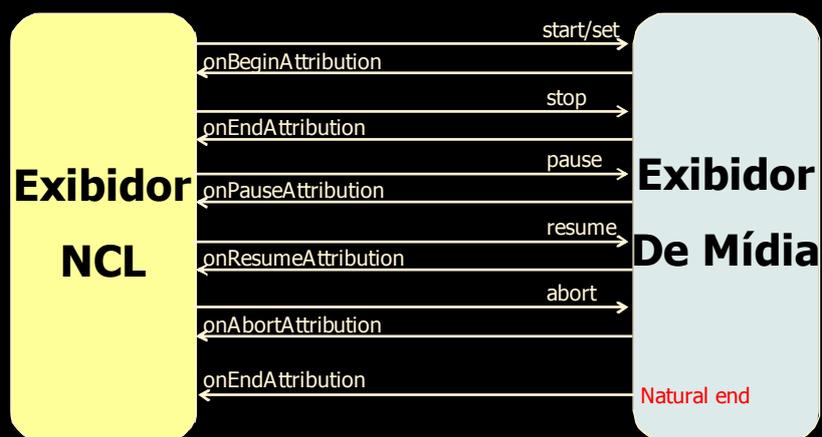
Eventos de Apresentação



Copyright © 2012 TeleMídia



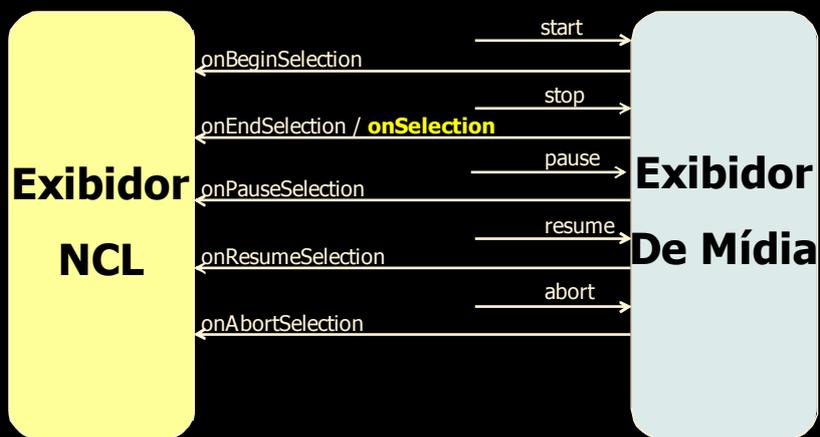
Eventos de Atribuição



Copyright © 2012 TeleMídia



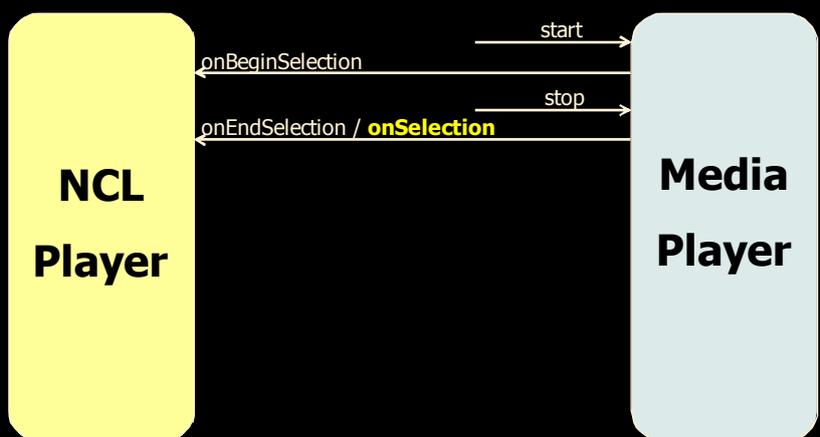
Eventos de Seleção



Copyright © 2012 TeleMídia



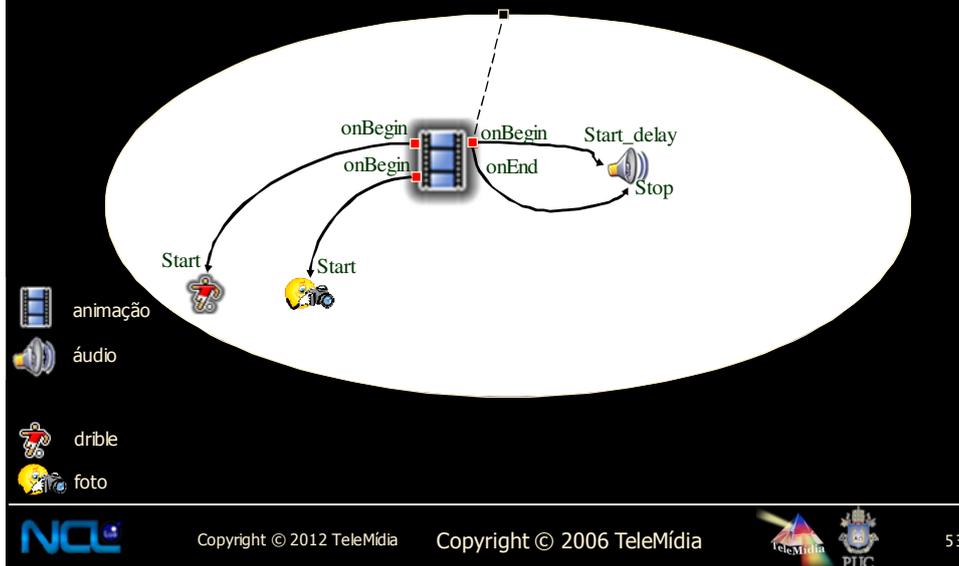
Eventos de Seleção



Copyright © 2012 TeleMídia



Exemplo 1 – Visão Estrutural



Connector *onEndStop*

```

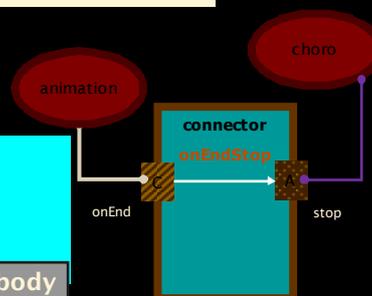
<causalConnector id="onEndStop">
  <simpleCondition role="onEnd"/>
  <simpleAction role="stop" max="unbounded" qualifier="seq" />
</causalConnector>
  
```

Elo que utiliza o conector *onBeginStart*:
Terminando o áudio *choro*

```

<link xconnector="onEndStop">
  bind role="onEnd" component="animation"/>
  <bind role="stop" component="choro" />
</link>
  
```

na seção body



Exemplo 1

```
<link id="IMusic" xconnector="onBeginStart_delay">
  <bind role="onBegin" component="animation" />
  <bind role="start" component="choro">
    <bindParam name="xyz" value="5s"/>
  </bind>
</link>

<link id="IDrible" xconnector="onBeginStart">
  <bind role="onBegin" component="animation" interface="segDrible"/>
  <bind role="start" component="drible"/>
</link>

<link id="IPhoto" xconnector="onBeginStart">
  <bind role="onBegin" component="animation" interface="segPhoto"/>
  <bind role="start" component="photo"/>
</link>

<link id="IEnd" xconnector="onEndStop">
  <bind role="onEnd" component="animation"/>
  <bind role="stop" component="choro"/>
</link>

</body>
</ncl>
```



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Exemplo de base de conectores -->
<ncl id="causalConnBase" xmlns="http://www.ncl.org.br/NCL3.0/causalConnectorProfile">
  <head>
    <connectorBase>
      <causalConnector id="onBeginStart_delay">
        <connectorParam name="delay"/>
        <simpleCondition role="onBegin"/>
        <simpleAction role="start" delay="$delay" max="unbounded" qualifier="par"/>
      </causalConnector>
      <causalConnector id="onBeginStart">
        <simpleCondition role="onBegin"/>
        <simpleAction role="start" max="unbounded" qualifier="par"/>
      </causalConnector>
      <causalConnector id="onEndStop">
        <simpleCondition role="onEnd"/>
        <simpleAction role="stop" max="unbounded" qualifier="par"/>
      </causalConnector>
    </connectorBase>
  </head>
```



```

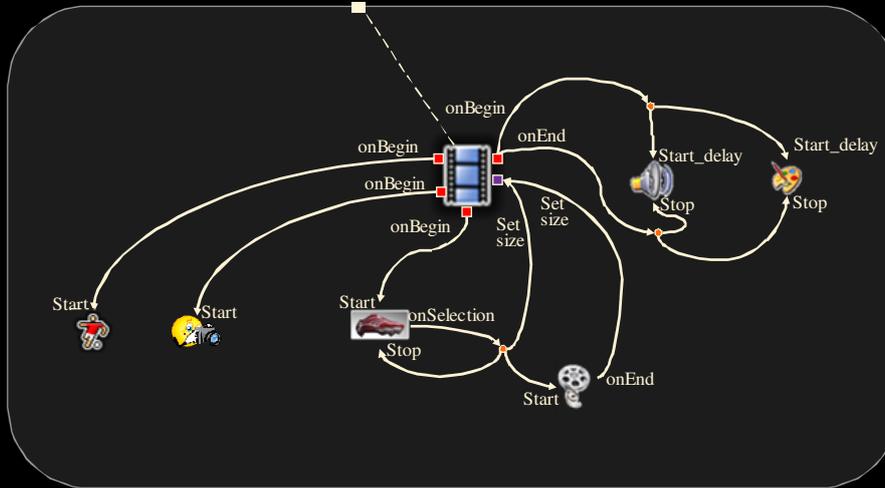
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Exemplo de sincronismo sem a interacao do usuario -->
<ncl id="sync" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
  <head>
    <regionBase>
      <region id="screenReg" height="100%" width="100%" zIndex="1">
        <region id="frameReg" left="5%" top="6.7%" height="18.5%" width="18.5%"
          zIndex="3"/>
      </region>
    </regionBase>
    <descriptorBase>
      <descriptor id="screenDesc" region="screenReg"/>
      <descriptor id="photoDesc" region="frameReg" explicitDur="5s"/>
      <descriptor id="audioDesc"/>
      <descriptor id="dribbleDesc" region="frameReg"/>
    </descriptorBase>
    <connectorBase>
      <importBase documentURI="causa1ConnBase.ncl" alias="conEx"/>
    </connectorBase>
  </head>

```

Exemplo 3 – Storyboard



Exemplo 3



Exemplo

```

<link id="IMusic" xconnector="conEx#onBeginStart_delay">
  <bind role="onBegin" component="animation"/>
  <bind role="start" component="background">
    <bindParam name="delay" value="5s"/>
  </bind>
  <bind role="start" component="choro">
    <bindParam name="delay" value="5s"/>
  </bind>
</link>

<link id="End" xconnector="conEx#onEndStop">
  <bind role="onEnd" component="animation"/>
  <bind role="stop" component="background"/>
  <bind role="stop" component="choro"/>
</link>

```

Exemplo 3

```
<media id="animation" src="media/animGar.mp4" descriptor="screenDesc">  
  <area id="segDrible" begin="12s"/>  
  <area id="segPhoto" begin="41s"/>  
  <area id="segIcon" begin="45s" end="51s"/>  
  
</media>
```

Exemplo 3

```
<link id="Icon" xconnector="conEx#onBeginStart">  
  <bind role="onBegin" component="animation" interface="segIcon"/>  
  <bind role="start" component="icon"/>  
</link>
```

Conector onKeySelectionStopSet_varStart

```
<causalConnector id="onKeySelectionStopSet_varStart">
  <connectorParam name="var"/>
  <connectorParam name="keyCode"/>
  <simpleCondition role="onSelection" key="$keyCode"/>
  <compoundAction operator="seq">
    <simpleAction role="stop" max="unbounded" qualifier="seq"/>
    <simpleAction role="set" value="$var"/>
    <simpleAction role="start" max="unbounded" qualifier="seq"/>
  </compoundAction>
</causalConnector>
```



Exemplo

```
<link id="IA advert" xconnector="conEx#onKeySelectionStopSet_varStart">
  <bind role="onSelection" component="icon">
    <bindParam name="keyCode" value="RED"/>
  </bind>
  <bind role="set" component="animation" interface="bounds">
    <bindParam name="var" value="5%,6.67%,45%,45%"/>
  </bind>
  <bind role="start" component="shoes"/>
  <bind role="stop" component="icon"/>
</link>
```



Exemplo 3

```
<media id="animation" src="media/animGar.mp4" descriptor="screenDesc">  
  <area id="segDrible" begin="12s"/>  
  <area id="segPhoto" begin="41s"/>  
  <area id="segIcon" begin="45s" end="51s"/>  
  <property name="bounds"/>  
</media>
```

Exemplo 3

```
<link id="IEndAdvert" xconnector="conEx#onEndSet_var">  
  <bind role="onEnd" component="shoes"/>  
  <bind role="set" component="animation" interface="bounds">  
    <bindParam name="var" value="0,0,100%,100%"/>  
  </bind>  
</link>
```

Exemplo 4 – Storyboard

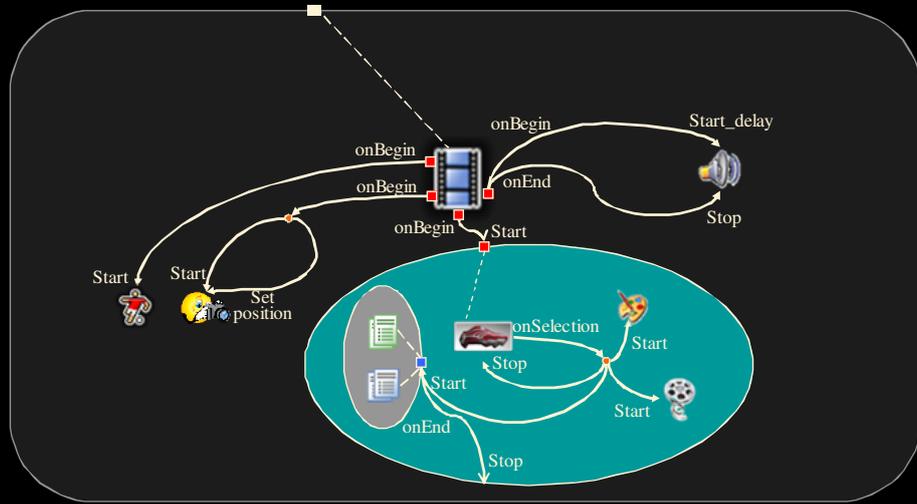


Exemplo 4 – Storyboard

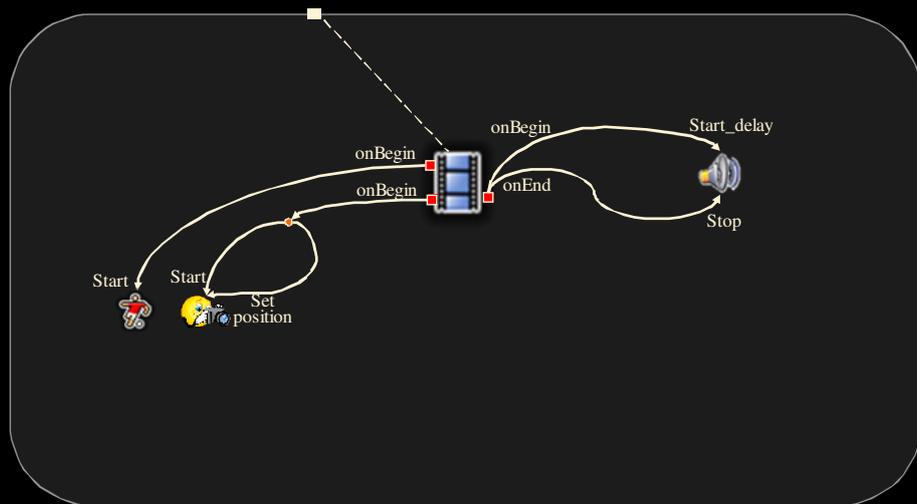




Exemplo 5



Exemplo 5

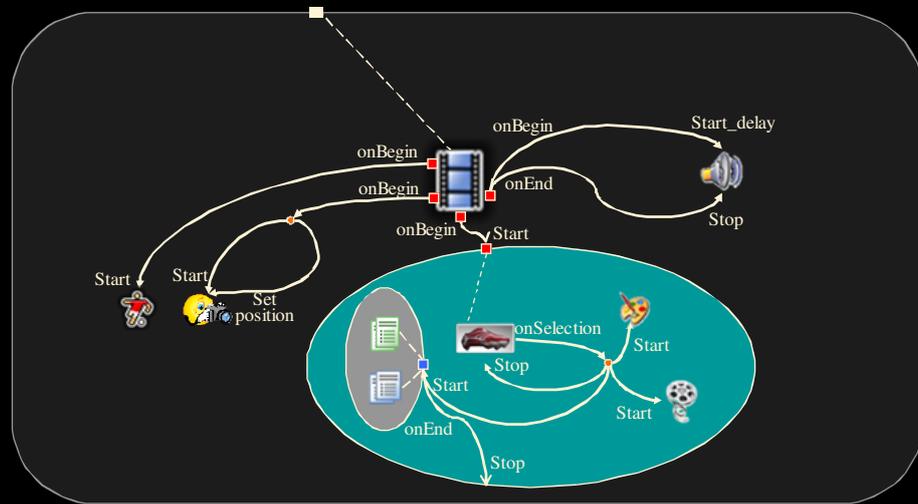


Importação

```
<head>  
  <importedDocumentBase>  
    <importNCL documentURI="Advert/advert.nc" alias="advert"/>  
  </importedDocumentBase>  
  ...  
</head>  
<body>  
  ...  
  <context id="advert" refer="advert#adv"/>  
  ...  
</body>
```

Objeto de Mídia NCL Embutido

Exemplo 6



Interfaces

- `<area id="a1" label="porta interna do body">`
- `<property name="porta interna do body">`

NCL Embutido

```
<head>
<regionBase>
  <region id="passiveAdvReg" left="5%" top="5%" width="40%" height="40%" zIndex="2"/>
  ...
</regionBase>
<descriptorBase>
  <descriptor id="passiveAdvDesc" region="passiveAdvReg" focusIndex="5" focusBorderWidth="0">
    <descriptorParam name="transparency" value="50%"/>
  </descriptor>
</descriptorBase>
...
</head>
<body>
...
  <media id="advert" src="Advert/advert.nc" descriptor="passiveAdvDesc"/>
...
</body>
```



Copyright © 2012 TeleMídia



81

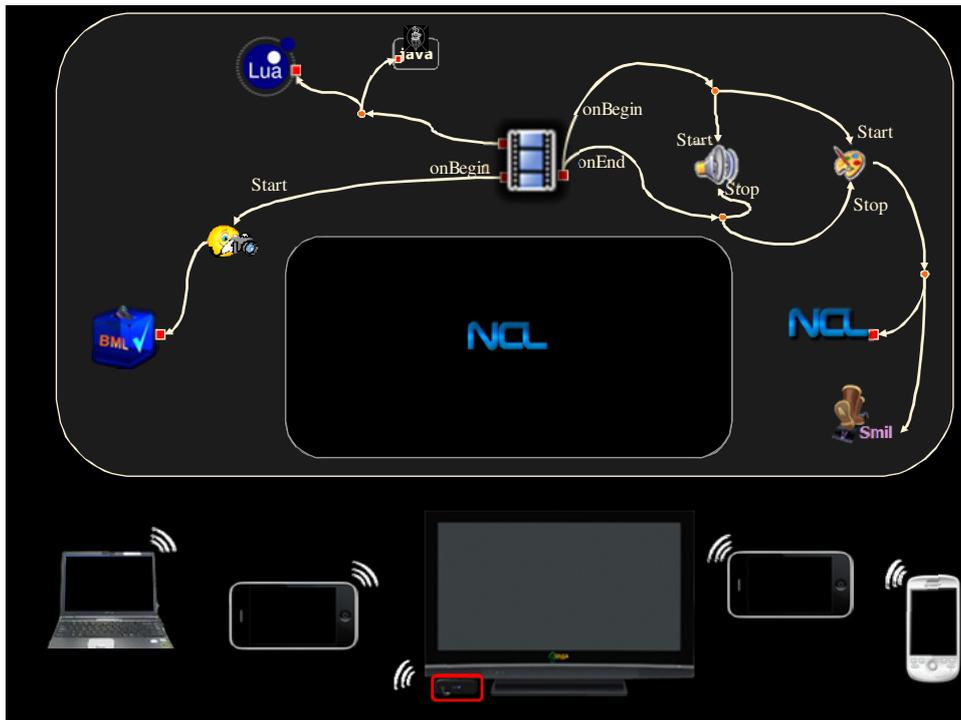
Múltiplos Dispositivos de Exibição

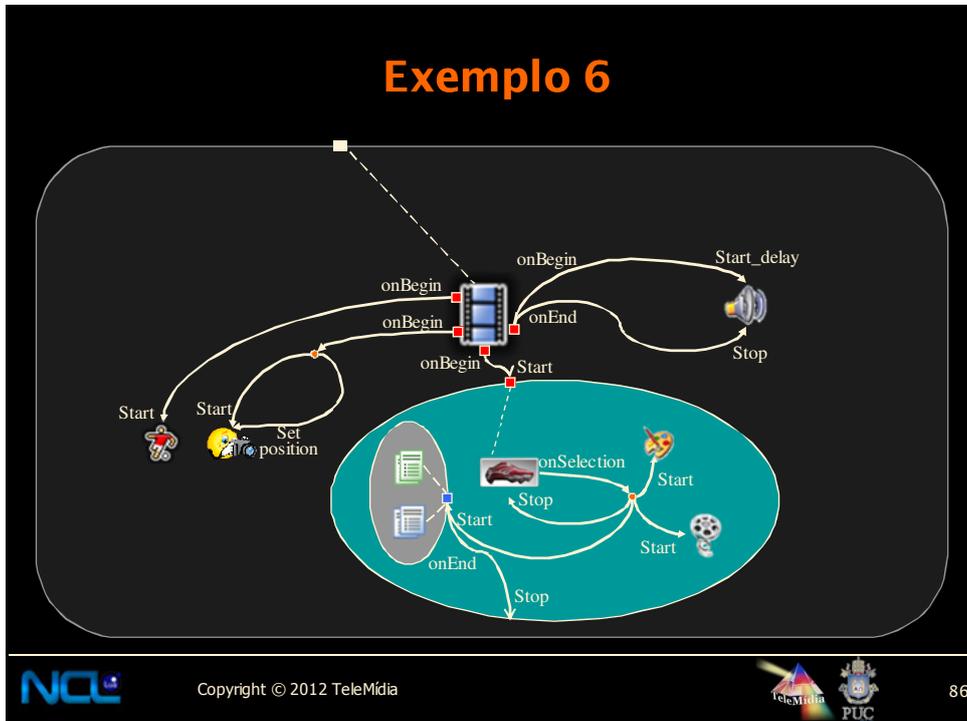


Copyright © 2012 TeleMídia

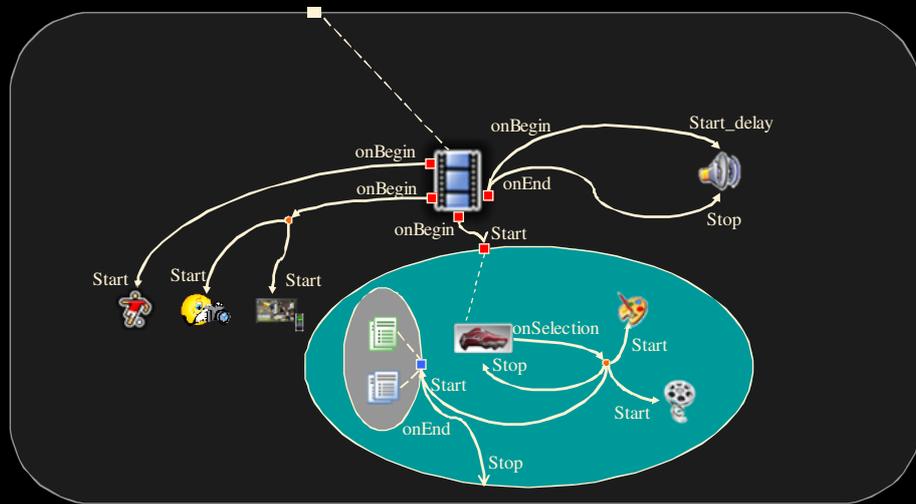


82





Exemplo 7



Classe Passiva

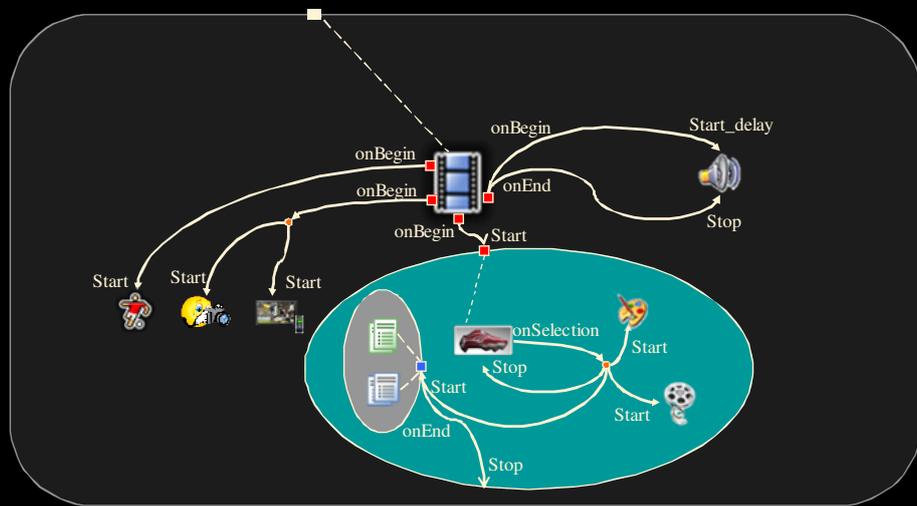
```

<head>
  <regionBase>
    ...
  </regionBase>
  <regionBase device="systemScreen(1)">
    <region id="frameReg2" width="100%" height="100%" zIndex="2"/>
  </regionBase>
  <descriptorBase>
    ...
    <descriptor id="photoDesc" region="frameReg2" explicitDur="6s"/>
  </descriptorBase>
  ...
</head>
<body>
  ...
  <media id="photo" src="media/photo.png" descriptor="photoDesc"/>
  ...
</body>

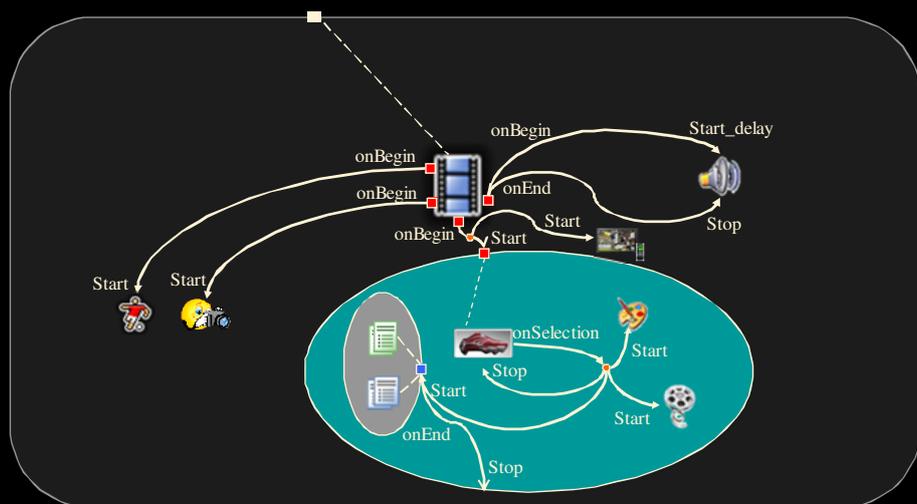
```



Exemplo 7



Exemplo 8



Classe Passiva

```
<head>
  <regionBase>
  ...
  </regionBase>
  <regionBase device="systemScreen(1)">
    <region id="passiveAdvReg" width="100%" height="100%" zIndex="1"/>
  </regionBase>
  <descriptorBase>
    <descriptor id="passiveAdvDesc" region="passiveAdvReg" focusIndex="5"
      focusBorderWidth="0">
      <descriptorParam name="transparency" value="50%"/>
    </descriptor>
  </descriptorBase>
</head>
<body>
  ...
  <media id="advert" src="Avert/advert.ncl" descriptor="passiveAdvDesc"/>
  ...
</body>
```



Copyright © 2012 TeleMídia



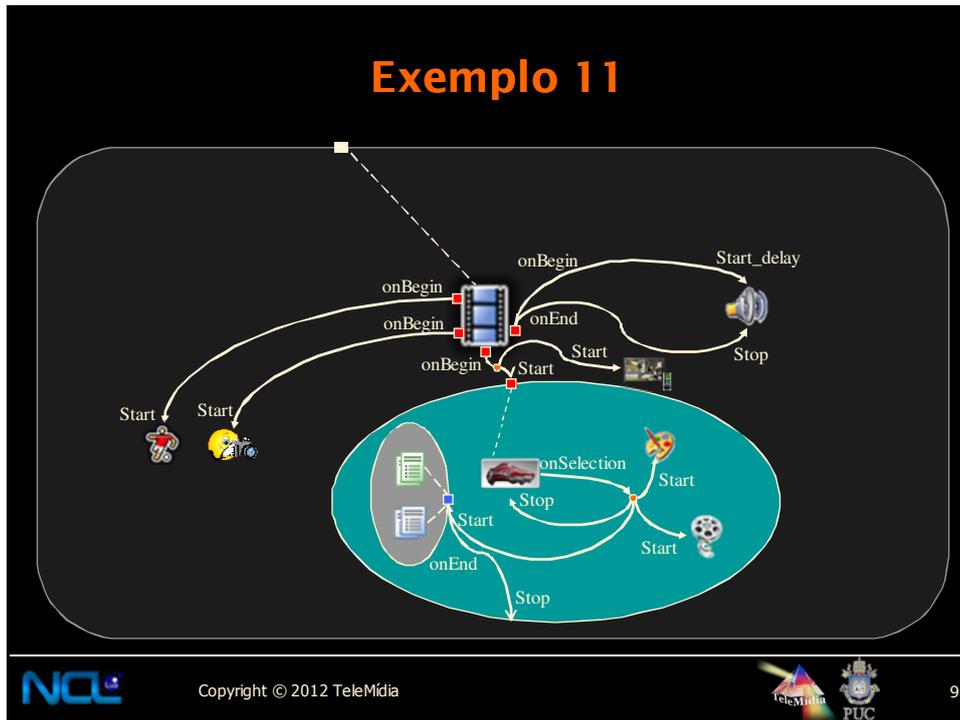
93



Copyright © 2012 TeleMídia



Exemplo 11



```

<head>
  <ruleBase>
    <rule id="single" var="system.devNumber(2)" value="0" comparator="eq"/>
  </ruleBase>
  <regionBase>
    ...
    <region id="NCLAdvertSingleReg" left="5%" top="5%" width="40%" height="40%" zIndex="2"/>
  </regionBase>
  <regionBase device="systemScreen(2)">
    <region id="NCLAdvertMultiReg" width="100%" height="100%" zIndex="1"/>
  </regionBase>
  <descriptorBase>
    ...
    <descriptorSwitch id="NCLAdvertDesc">
      <bindRule constituent="NCLAdvertSingleDesc" rule="single"/>
      <defaultDescriptor descriptor="NCLAdvertMultiDesc"/>
      <descriptor id="NCLAdvertSingleDesc" region="NCLAdvertSingleReg" focusIndex="2"/>
      <descriptor id="NCLAdvertMultiDesc" region="NCLAdvertMultiReg"/>
    </descriptorSwitch>
  </descriptorBase>
  ...
</head>

```

Classe Ativa

Plug-ins



Copyright © 2012 TeleMídia



101

Plug-ins

- Principal utilidade: exibir conteúdo cujo formato não é tratado pelo exibidor da linguagem
 - conteúdos específicos das aplicações (PDF, PostScript, etc.), áudio, vídeo
- Permite também que aplicações existentes sejam facilmente integradas à linguagem mãe (exemplo: plugin Ginga-NCL para HTML)



Copyright © 2012 TeleMídia



Plug-ins HTML



Copyright © 2012 TeleMídia



103

Plug-ins

- Tecnologia originalmente projetada pela Netscape
 - Netscape Navigator 2.0
 - Internet Explorer 3.0 passou também a oferecer suporte
- Módulo de código separado que se comporta como se fosse parte do browser
 - associado a um ou mais tipos de mídia (tipo MIME)
 - biblioteca de código nativo C
 - específico a uma plataforma (sistema operacional)
 - dependente da interface de programação do browser



Copyright © 2012 TeleMídia



Como inserir plug-ins em páginas HTML

- Elementos HTML utilizados para inserção de plug-ins
 - *OBJECT*
 - *EMBED* (não faz parte da especificação HTML 4.01)
- Modos de exibição de um plug-in
 - embutido
 - escondido
 - página inteira



Copyright © 2012 TeleMídia



Como inserir plug-ins em páginas HTML

- Elemento *OBJECT*
 - quando o browser não sabe tratar a especificação, o conteúdo do elemento deve ser apresentado
 - *Objects* podem ser aninhados

```
<object data="clock.avi" type="video/msvideo" height="100%" width="100%"  
classid="http://microsoft.com/plugins/" >  
  < object data="clock.gif" type="image/gif">  
  <p>Hora certa.  
  </object>  
</object>
```



Copyright © 2012 TeleMídia



Como inserir plug-ins em páginas HTML

- Principais atributos do elemento **OBJECT**
 - *data*: URI do conteúdo a ser tratado pelo plug-in
 - *type*: tipo MIME do dado
 - *classid*: URI de onde encontrar um plug-in a ser baixado caso o browser não tenha o plug-in instalado
 - *height* e *width*: tamanho da área para exibição
 - *name* ou *id*: identificação para o plug-in



Copyright © 2012 TeleMídia



Elemento Object

- Container genérico para embutir conteúdo externo em páginas HTML (não se restringe a plug-ins)
 - imagens, applets, plug-ins, documentos html embutidos, etc.
 - deve substituir o elemento APPLET do HTML
 - forma genérica para inclusão de imagens (elemento IMG) e outros documentos HTML (elemento IFRAME)
- Parâmetros podem ser passados para um OBJECT através do uso do elemento PARAM
 - `<param name="nome_parâmetro" valuetype="tipo" value="valor" type="mime_content_type">`



Copyright © 2012 TeleMídia



Plug-ins - Modelo de Execução

- Plug-ins executam no mesmo espaço de memória do browser
 - DLLs, objetos compartilhados, bibliotecas compartilhadas, etc.
- Ciclo de vida de um plug-in está associado ao ciclo de vida da página que o aciona



Copyright © 2012 TeleMídia



Plug-ins - Modelo de Execução

- Quando o browser encontra em uma página uma referência (URI) para um arquivo que está associado a um Plug-in
 - browser carrega o código do plug-in na memória (se ainda não tiver feito)
 - cria uma nova instância do plug-in (o browser pode criar várias instâncias de um mesmo plug-in simultaneamente)



Copyright © 2012 TeleMídia



Plug-ins - Modelo de Execução

- Quando o browser sai da página que contém a referência para o plug-in ou tem sua janela fechada, a instância do plug-in é removida da memória
 - quando a última instância de um plug-in é removida, o código do plug-in é retirado da memória
- Quando um plug-in não está carregado em memória, o mesmo só ocupa espaço em disco



Copyright © 2012 TeleMídia



Plug-ins HTML

- Dependentes de plataforma e browser
- Não permite interagir diretamente com o conteúdo HTML
 - substituir imagens (simular animações)
 - simular menus de opções
 - mudar características de apresentação do documento de acordo com a interação do usuário
 - acrescentar conteúdo dinamicamente



Copyright © 2012 TeleMídia



Plug-ins NCL



Copyright © 2012 TeleMídia



113

Media Player API

Implemented by:	Operation (input parameters)
Media player	prepare (mediaType media)
Host NCL Player	notifyAudioBuffer (any buffer)
Host NCL Player	notifyVideoBuffer (any buffer)
Media player	start (ID eventId)
Media player	addEvent (event Type event)
Media player	removeEvent (ID eventId)
Media player	stop (ID eventId)
Media player	abort (ID eventId)
Media player	pause (ID eventId)
Media player	resume (ID eventId)
Host NCL Player	notifyEventTransition (ID eventId, transitionType transition)
Media player	requestPropertyValue(string name)
Host NCL Player	notifyPropertyValue (string name, string value)
Media player	setPropertyValue(string name, string value, string duration, string by)
Host NCL Player	notifyError(string message)



Copyright © 2012 TeleMídia



114

Input Control API

Implemented by:	Operation (input parameters)
Host NCL Player	notifyInputControl ()
Media player	requestInputControl (deviceType device, keyListType keyList)
Host NCL Player	notifyInput (deviceType device, sensorType sensor)
Media player	nestInput (unsignedInteger nestingLevel)



Copyright © 2012 TeleMídia



115

Plug-ins NCL

- Dependentes de plataforma e browser
- **ão permite** interagir diretamente com o conteúdo , por meio de elos
 - substituir imagens (simular animações)
 - simular menus de opções
 - mudar características de apresentação do documento de acordo com a interação do usuário
 - acrescentar conteúdo dinamicamente



Copyright © 2012 TeleMídia



Tecnologias para Processamento no Lado do Cliente WWW



Copyright © 2012 TeleMídia



NCL

Objetos Imperativos Embutidos



Copyright © 2012 TeleMídia



118

Interfaces

- `<area id="a1" label="função interna">`
- `<property name="função interna" value="parametro">`



Copyright © 2012 TeleMídia



Objetos NCLua Embutidos



Copyright © 2012 TeleMídia



120

Por que Linguagens de Script?

- Interpretadas (não exigem compilação)
 - independência de plataforma
- Simples de programar
 - mais adequadas para usuários não experts em programação
 - ideais para tarefas simples
- Desvantagens
 - Ineficiência (interpretadas)
 - menos poderosas e por isso são indicadas apenas para tarefas não muito complexas



Copyright © 2012 TeleMídia



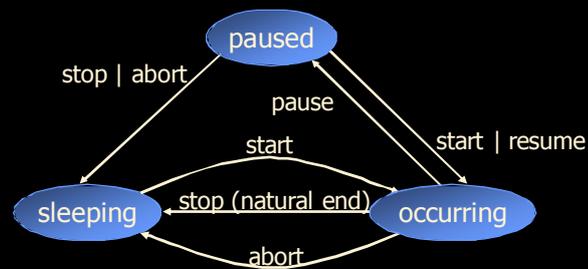
```
local counter = 0
local dx, dy = canvas:attrSize()    -- dimensoes do canvas

function handler1 (evt)
  if evt.class=='ncl' and evt.type=='attribution' and evt.action=='start' and evt.name=='add' then
    counter = counter + evt.value
  end
end

function handler2 (evt)
  canvas:attrColor ('black')
  canvas:drawRect('fill',0,0,dx,dy)
  canvas:attrColor ('yellow')
  canvas:attrFont ('vera', 24, 'bold')
  canvas:drawText (10,10, 'O número de vezes que você trocou de ritmo foi: '..counter)
  canvas:flush()
end

event.register(handler1)
event.register(handler2,'ncl','presentation','fim','start')
```

Máquina de Estado de Evento



Copyright © 2012 TeleMídia



```
local counter = 0
local dx, dy = canvas:attrSize()    -- dimensoes do canvas

function handler1 (evt)
  if evt.class=='ncl' and evt.type=='attribution' and evt.action=='start' and evt.name=='add' then
    counter = counter + evt.value
    event.post {
      class = 'ncl',
      type = 'attribution',
      name = 'add',
      action = 'stop',
      value = counter,
    }
  end
end

function handler2 (evt)
  canvas:attrColor ('black')
  canvas:drawRect('fill',0,0,dx,dy)
  canvas:attrColor ('yellow')
  canvas:attrFont ('vera', 24, 'bold')
  canvas:drawText (10,10, 'O número de vezes que você trocou de ritmo foi: '..counter)
  canvas:flush()
  event.post {
    class = 'ncl',
    type = 'presentation',
    label = 'fim',
    action = 'stop',
  }
end

event.register(handler1)
event.register(handler2,'ncl','presentation','fim','start')
```

HTML

Objetos Imperativos embutidos



Copyright © 2012 TeleMídia



125

Scripts



Copyright © 2012 TeleMídia



126

Scripts

- Utilização de uma linguagem de script para adicionar funcionalidades dinâmicas a páginas HTML estáticas
- Página HTML carrega (de forma embutida ou através de uma referência) scripts que serão executados pelo browser
 - alterar a especificação de apresentação dos elementos: **adaptação**
 - acrescentar conteúdo dinamicamente ao documento: **sincronização**
 - verificar a entrada de dados em um formulário
 - controlar o browser



Copyright © 2012 TeleMídia



Linguagens de Script

- Principais linguagens de script utilizadas
 - JavaScript (inicialmente chamado de LiveScript) - Netscape
 - Jscript e VBScript - Microsoft
 - Tcl
- Padrão para linguagens de script interpretadas no cliente
 - ECMAScript (European Computer Manufacturers Association Script)
 - padrão de junho de 1997
 - JavaScript e JScript são implementações do padrão



Copyright © 2012 TeleMídia



ECMAScript

- Padrão para linguagens de script interpretadas no cliente
- Linguagem padrão dos middlewares declarativos
 - DVB-HTML
 - ACAP-X
 - BML
 - HbbTV
- Linguagem também suportada no Ginga-NCL, onde a linguagem script padrão é Lua.

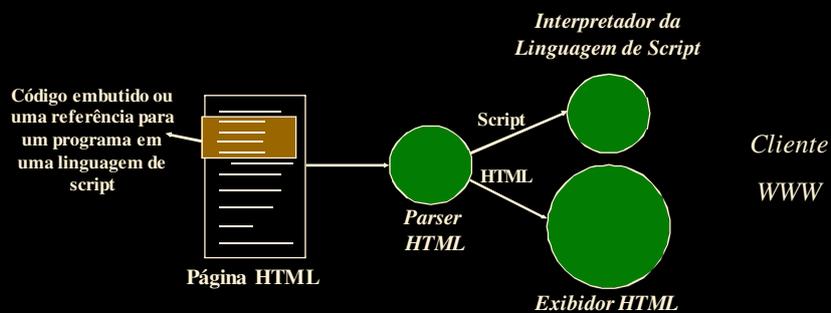


Copyright © 2012 TeleMídia



Scripts

- Para executar os scripts, o cliente WWW (browser) precisa de um interpretador da linguagem de script utilizada no documento



Copyright © 2012 TeleMídia



Como inserir Scripts em páginas HTML

- Elemento **SCRIPT**
 - **src**: URI para o script propriamente dito
 - se não for definido, o conteúdo do elemento será tratado como sendo o código
 - **type**: especifica a linguagem de script através de um tipo MIME
 - ex: text/javascript
 - **charset**: apenas considerado quando o atributo *src* for definido
 - **language**: desaconselhado, substituído pelo atributo *type*



Copyright © 2012 TeleMídia



Como inserir Scripts em páginas HTML

- Exemplo de **SCRIPT** em JavaScript

```
<p>Última atualização feita em:  
<script type="text/javascript">  
<!--  
document.write(document.lastModified);  
-->  
</script>
```

Comentários do HTML:

para evitar que browsers que não dão suporte a scripts exibam o conteúdo do script na tela

Informa a data da última modificação do documento



Copyright © 2012 TeleMídia



Como inserir Scripts em páginas HTML

- Elemento pode aparecer várias vezes, tanto no *Head* como no *Body* do documento HTML
- Fica a cargo de cada linguagem de script oferecer uma sintaxe para referenciar elementos HTML no documento



Copyright © 2012 TeleMídia



Como inserir Scripts em páginas HTML

- Eventos intrínsecos: atributos de elementos HTML que permitem a definição de scripts para tratarem eventos na apresentação do documento
 - Body e Frameset
 - onload, onunload
 - A, Area, Label, Input, Select, Textarea e Button
 - onfocus, onblur
 - Form
 - onsubmit, onreset



Copyright © 2012 TeleMídia



Como inserir Scripts em páginas HTML

- Eventos intrínsecos (continuação)
 - Input e Textarea
 - onselect
 - Input, Textarea e Select
 - onchange
 - com a maioria dos elementos HTML
 - onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onkeypress, onkeydown, onkeyup



Copyright © 2012 TeleMídia



Exemplo de Script Associado a Eventos HTML

```
<form>  
<input type="button" value="Aperte aqui!"  
      onclick="alert('Você clicou no botão!')">  
</form>
```

- Exibe uma mensagem quando o usuário pressiona o botão
- Os eventos intrínsecos normalmente são utilizados em conjunto com funções declaradas na área de SCRIPT



Copyright © 2012 TeleMídia



Exemplo de Script Associado a Eventos HTML

```
<head>
  <script type="text/javascript">
    function my_onclick() {
      ...
    }
  </script>
</head>
<body>
  ...
  <form>
    <button type="button" name="mybutton" onclick="my_onclick()">
      mybutton
    </button>
  </form>
</body>
```



Copyright © 2012 TeleMídia



Exemplo de Script Associado a Eventos HTML

```
<form name="form">
  <input type="button" name="mybutton" value="mybutton">
  <script type="text/javascript">
    function my_onclick() {
      ...
    }
    document.form.mybutton.onclick = my_onclick()
  </script>
</input>
</form>
```



Copyright © 2012 TeleMídia



Scripts (Exemplo DHTML)

- Sublinha a âncora só quando o mouse estiver sobre a mesma

```
<html><head>
  <style>
    .on {text-decoration:underline; color:blue;}
    .off {text-decoration:none; color:black;}
  </style>
</head><body>
  <ul>
    <li> <a href="apresentacoes.html" class="off"
      onmouseover="this.className='on';"
      onmouseout="this.className='off';">Apresentações</a>
    <li> <a href="artigos.html" class="off"
      onmouseover="this.className='on';"
      onmouseout="this.className='off';">Artigos</a>
  </ul>
</body></html>
```



Copyright © 2012 TeleMídia



O que pode ser feito com Scripts

- Acrescentar dinamicamente conteúdo a uma página HTML enquanto a página está sendo processada pelo browser (**sincronismo**)
- Alterar o conteúdo de alguns elementos HTML (**adaptação de conteúdo**)
- Alterar características de apresentação dos elementos (**adaptação de apresentação**)
- Obter a lista de elos do documento



Copyright © 2012 TeleMídia



O que pode ser feito com Scripts

- Controlar o comportamento do browser
 - exibir mensagens através de dialog boxes ou na linha de status do browser
 - abrir e fechar novas janelas do browser
 - navegar no histórico de documentos simulando o comportamento dos botões back e forward do browser
- Ler e escrever valores em elementos para entradas de dados em qualquer formulário do documento
- Ler e escrever em Cookies



Copyright © 2012 TeleMídia



O que NÃO pode ser feito com Scripts

- Linguagens não possuem recursos gráficos
- Por razões de segurança, não se tem acesso para ler ou escrever em arquivos no cliente
- Não oferece suporte para comunicação em rede a não ser baixar o conteúdo de uma URL
- Não tem capacidade para abrir múltiplas threads



Copyright © 2012 TeleMídia



ECMAScript em iTV

- Muitas das aplicações para TV digital baseadas em ECMAScript fazem parte do contexto de operação de DHTML (*Dynamic HTML*), permitindo converter uma página HTML estática em uma aplicação interativa.



Copyright © 2012 TeleMídia



Linguagens de Sistemas



Copyright © 2012 TeleMídia



144

Linguagens de Sistema

- Outra possibilidade para introduzir dinamismo em documentos na Web é através de uma linguagem de sistema
 - programas precisam ser distribuídos e executados em uma variedade de plataformas (clientes WWW)
 - programa (compilado) precisa ser independente de plataforma
- Java
 - linguagem orientada a objetos, independente de plataforma
 - desenvolvida pela Sun Microsystems

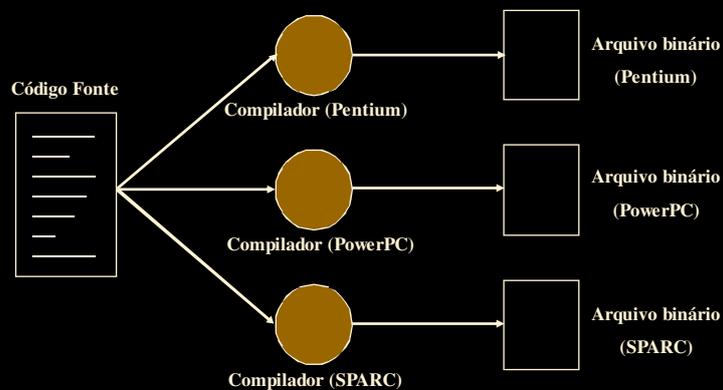


Copyright © 2012 TeleMídia



Linguagens de Sistema

- Programas compilados tradicionais (dependentes de plataforma)

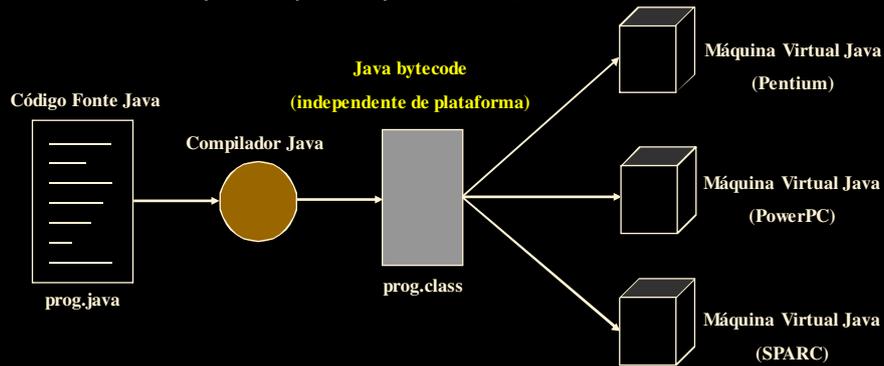


Copyright © 2012 TeleMídia



Linguagens de Sistema

- Programas Java (independentes de plataforma)
 - desvantagem: desempenho => bytecode precisa ser interpretado pela máquina virtual Java



Applets Java

Applets Java

- Programas que podem ser baixados de qualquer servidor WWW
- Executados localmente em qualquer cliente WWW que saiba executar código Java
 - browsers possuem uma máquina virtual Java (JVM) embutida ou implementada como um plug-in
- Quando o browser carrega uma página Web que faz referência a um applet, ele traz o applet de um servidor WWW (bytecode) e o executa localmente



Copyright © 2012 TeleMídia



Applets Java

Documento HTML



Cliente

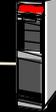
WWW

- Inicializa Máquina Virtual Java
- Executa applet

applet
bytecode



Rede



Servidor

WWW

Documento HTML



applet
bytecode



Copyright © 2012 TeleMídia



O que é um Applet

- Uma classe Java que estende a **classe Applet**
 - pacote (package) java.applet
- Principais métodos definidos pela classe
 - public void init()
 - public void start()
 - public void stop()
 - public void destroy()
 - public void paint(Graphics g)
 - public String getParameter(String name)



Copyright © 2012 TeleMídia



Applet Java

```
import java.applet.*;
public class HelloAgain extends Applet {
    public void init() {
        System.out.println("init - Hello Again");}
    public void start() {
        System.out.println("start - Hello Again");}
    public void stop() {
        System.out.println("stop - Hello Again");}
    public void destroy() {
        System.out.println("destroy - Hello Again");}
}
```



Copyright © 2012 TeleMídia



Applet Java

```
import java.applet.*;
import java.awt.*;
public class HelloAgain extends Applet {
    private int cont;
    public void init() {
        cont= 0;}
    public void start() {
        cont= cont + 1;}
    public void paint(Graphics g) {
        g.drawString("Hello again: " + cont + "!", 5, 40);}
}
```



Copyright © 2012 TeleMídia



Como inserir Applets em páginas HTML

- Elemento **OBJECT**
 - *codebase*: URI base para localizar o applet
 - *codetype*: tipo MIME do objeto ("application/java")
 - *classid*: localização do applet ("java:<nome_classe>")
 - *height* e *width*: tamanho da área para exibição
 - *name* ou *id*: nome de identificação para o applet



Copyright © 2012 TeleMídia



Como inserir Applets em páginas HTML

```
<object codetype="application/java"  
      classid="java:HelloAgain" width="100"  
      height="100">
```

Texto caso o browser não trate applets.

```
</object>
```

```
<object codetype="application/java" codebase="/applets/"  
      classid="java:HelloAgain" width="100" height="100">
```

Texto caso o browser não trate applets.

```
</object>
```



Copyright © 2012 TeleMídia



Applets e Parâmetros

```
<object codetype="application/java"  
      classid="java:HelloAgain" width="100" height="100">
```

```
<param name="user" value="LFGS">
```

Texto caso o browser não trate applets.

```
</object>
```



Copyright © 2012 TeleMídia



Applets e Parâmetros

```
import java.applet.*;
import java.awt.*;

public class HelloAgain extends Applet {
    private String user;
    public void init() {
        user= getParameter("user");
    }
    public void paint(Graphics g) {
        g.drawString("Hello again: " + user);
    }
}
```



Copyright © 2012 TeleMídia



O que NÃO é permitido com Applets Java

- Ler ou escrever no sistema de arquivos local do cliente
- Rodar programas executáveis no cliente
- Efetuar chamadas a métodos nativos
- Comunicar-se com um servidor WWW que não seja o servidor de onde foi feito download do applet (*originating host*)
- Consultar propriedades do sistema, exceto a versão da máquina virtual Java e nome/versão do sistema operacional



Copyright © 2012 TeleMídia



Integração das Tecnologias

- Como fazer a integração entre elementos HTML, scripts, applets e plug-ins?
 - depende do browser
 - Netscape => *LiveConnect*
 - Internet Explorer => *ActiveX*
- Permitem:
 - chamar métodos de applets e plug-ins a partir de JavaScript
 - chamar funções JavaScript a partir de Java
 - chamar métodos Java a partir de plug-ins



Copyright © 2012 TeleMídia



Processamento no Cliente

- Vantagens
 - distribuição do processamento
 - tempo de resposta menor, dependendo da aplicação
 - simplifica a implementação dos servidores
- Desvantagens
 - aumenta os riscos de não portabilidade
 - aumenta a complexidade dos clientes
 - requer restrições de segurança
 - ineficiente para programação utilizando bases de dados no servidor



Copyright © 2012 TeleMídia



Xlets Java



Copyright © 2012 TeleMídia



161

Xlet

- A API JavaTV introduz o conceito de *xlets*, que é em linhas gerais bastante similar a *applets*. Assim como *applets*, *xlets* têm seu ciclo de vida controlado por algum elemento presente no ambiente de execução. No caso de *applets*, esse ambiente é tipicamente um navegador. No caso de *xlets*, há um *gerente de aplicações* residente no terminal de acesso.
- No entanto, há duas diferenças principais entre *applets* e *xlets*.
 - Em primeiro lugar, *xlets* podem ser suspensos (*paused*) e reiniciados (*resumed*). Como em um terminal de acesso várias aplicações podem estar executando concorrentemente, mas em geral somente uma delas pode estar visível em um dado instante, a suspensão de uma aplicação que não esteja visível permite que ela não consuma recursos do terminal de acesso, que podem então ser utilizados pela aplicação visível.
 - A segunda diferença é que *xlets* têm uma capacidade de interação com o seu ambiente de execução muito mais limitada que *applets*. No entanto, tais interações podem ser conseguidas indiretamente por meio de outras APIs.



Copyright © 2012 TeleMídia

