

Interacção e Animação

Interacção: a acção ocorre em resposta a estímulos provocados pelo utilizador

Animação: a acção ocorre pela passagem do tempo

Behavior class: classe abstracta que fornece os mecanismos necessários para responder a eventos possibilitando a alteração do grafo em *run time*

Interacção e Animação

Exemplos de estímulos: teclado, rato, colisão de objectos, tempo, combinação de vários eventos,...

Exemplos de Acções: adicionar/remover objectos da cena, mudar atributos de objectos, lançar *Threads*,...

As subclasses de Behavior têm de definir:

Método **initialize()** - define o evento que activa esse behavior

Método **processStimulus(Enumeration c)** – método invocado pelo sistema quando ocorre o evento correspondente. A última instrução deve definir novamente a nova condição de activação.

Scheduling Region : especifica a região do espaço onde o **behavior** é válido. Restringe a região onde são verificadas as condições de activação. Melhora o desempenho do sistema.

Interacção e Animação

Tipos de *scheduling region*:

- BoundingSphere
- BoundingBox
- BoundingPolytope
 - permite definir regiões a partir de equações de planos, pela reunião das regiões definidas por um conjunto de objectos, etc

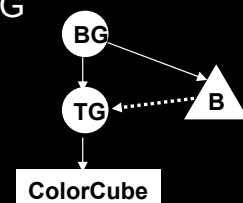
Utility Classes para Teclado/Rato

`com.sun.j3d.utils.behaviors.*`

- `KeyNavigatorBehavior(TransFormGroup tg)`
- `MouseBehavior()`
 - `MouseRotate()`
 - `MouseTranslate()`
 - `MouseZoom()`

Inclusão no programa:

1. Activar as *flags* de permissão de leitura/escrita do TG
2. Criar o objecto *Behavior*
3. Atribuir o TG alvo
4. Especificar a região de influência
5. Adicionar o *Behavior* ao grafo



Exemplo: teclado

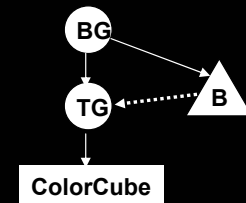
```
public class SimpleBehavior extends Behavior{

    private TransformGroup targetTG;
    private Transform3D rotation = new Transform3D();
    private double angle = 0.0;

    // create SimpleBehavior
    SimpleBehavior(TransformGroup targetTG){
        this.targetTG = targetTG;
        //targetTG.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    }

    // initialize the Behavior - set initial wakeup condition
    // called when behavior beacomes live
    public void initialize(){
        this.wakeupOn(new WakeupOnAWTEvent(KeyEvent.KEY_PRESSED));
    }

    // called by Java 3D when appropriate stimulus occurs
    public void processStimulus(Enumeration criteria){
        // decode event, do what is necessary
        angle += 0.1;
        rotation.rotY(angle);
        targetTG.setTransform(rotation);
        this.wakeupOn(new WakeupOnAWTEvent(KeyEvent.KEY_PRESSED));
    }
} // end of class SimpleBehavior
```



(ver exemplo: SimpleBehavior.java)

Descodificação

```
public void processStimulus (Enumeration criteria) {
    WakeupCriterion wakeup;
    AWTEvent[] event;
    int id, i;

    while (criteria.hasMoreElements()) {
        wakeup = (WakeupCriterion) criteria.nextElement();
        if (wakeup instanceof WakeupOnAWTEvent) {
            event = ((WakeupOnAWTEvent) wakeup).getAWTEvent();

            for(i=0; i < event.length; i++)
            {
                id = event[i].getID();
                if(id==KeyEvent.KEY_PRESSED) {
                    if (((KeyEvent) event[i]).GetKeyCode() == KeyEvent.VK_S){
                        // processamento do evento
                        targetTG.getTransform(transl);
                        transl.mul(offsetRight);
                        targetTG.setTransform(transl);
                    }
                }
            }
        }
    }
    this.wakeupOn(new WakeupOnAWTEvent(KeyEvent.KEY_PRESSED));
}
```

Subclasses de WakeupCriterion

WakeupOnActivation
→ WakeupOnAWTEvent
WakeupOnBehaviorPost
WakeupOnCollisionEntry
WakeupOnCollisionExit
WakeupOnCollisionMovement
WakeupOnDeactivation
→ WakeupOnElapsedFrames
→ WakeupOnElapsedTime
WakeupOnSensorEntry
WakeupOnSensorExit
WakeupOnTransformChange
WakeupOnViewPlatformEntry
WakeupOnViewPlatformExit

Combinação de eventos

WakeupAnd

WakeupAndOfOrs

WakeupOr

WakeupOrOfAnds

Exemplo: eventos do rato

```
public class MyMouseBehavior extends Behavior{
    WakeupCriterion[] mouseEvents;
    WakeupOr mouseCriterion;

    MyMouseBehavior() {}

    public void initialize(){
        mouseEvents = new WakeupCriterion[3];
        mouseEvents[0] = new WakeupOnAWTEvent(MouseEvent.MOUSE_DRAGGED);
        mouseEvents[1] = new WakeupOnAWTEvent(MouseEvent.MOUSE_PRESSED);
        mouseEvents[2] = new WakeupOnAWTEvent(MouseEvent.MOUSE_RELEASED);
        mouseCriterion = new WakeupOr(mouseEvents);
        wakeupOn(mouseCriterion);
    }

    public void processStimulus(Enumeration criteria){
        while (criteria.hasMoreElements()) {
            WakeupCriterion wakeup = (WakeupCriterion) criteria.nextElement();

            if (wakeup instanceof WakeupOnAWTEvent) {
                AWTEvent[] event = ((WakeupOnAWTEvent)wakeup).getAWTEvent();

                for (int i=0; i<event.length; i++) {
                    id = event[i].getID();

                    if (id == MouseEvent.MOUSE_DRAGGED) {}
                    else if(id == MouseEvent.MOUSE_PRESSED){}
                    else if(id == MouseEvent.MOUSE_RELEASED){}
                }
            }
            wakeupOn(mouseCriterion);
        }
    }
}
```

Seleccção de objectos na cena 3D (Picking)

JavaX.media.j3d.PickShape

- PickBounds
- PickCone
- PickCylinder
- PickPoint
- PickRay, PickSegment

Utilização:

1. Definir a *PickShape* a usar no teste de intercepção
2. Obter o caminho para os objectos seleccionados

```
SceneGraphPath[] BranchGroup.pickAll(PickShape pickShape)
```

```
SceneGraphPath[] pickAllSorted(PickShape pickShape)
```

```
SceneGraphPath pickClosest(PickShape pickShape)
```

```
SceneGraphPath pickAny(PickShape pickShape)
```

Seleccção de objectos na cena 3D (Picking)

Flag a activar nos nós que se quiserem incluir no vector:

Node.ALLOW_PICK_REPORTING

Nota: por defeito a *flag* assume o valor FALSE, i.e. por defeito o vector devolvido pela função está vazio.

Utility Classes para a Seleccção de objectos

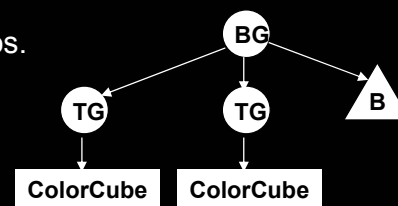
(classe abstracta)

`com.sun.j3d.utils.picking.behaviors.PickMouseBehavior`

- `PickRotateBehavior(objRoot, canvas, bounds)`
- `PickTranslateBehavior`
- `PickZoomBehavior`

Ver exemplo do tutorial: `MousePickApp.java`

- Apenas um behavior para os vários objectos.



Utility Classes para a Selecção de objectos

com.sun.j3d.utils.picking

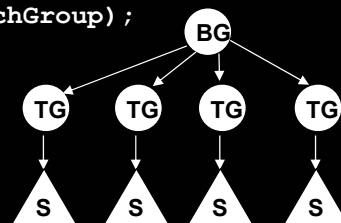
- `PickTool (BranchGroup bg)`
 - `PickCanvas (Canvas3D c, BranchGroup b)`
- `PickIntersection ()`
- `PickResult ()`

A classe `PickCanvas` permite simplificar a operação de selecção: utiliza a coordenada de posição do rato no canvas para criar a `PickShape` apropriada.

- `PickResult[] PickCanvas.pickAll ()`
- `PickResult PickCanvas.pickAny ()`
- `PickResult[] PickCanvas.pickAllSorted ()`
- `PickResult PickCanvas.pickClosest ()`

Exemplo de aplicação

```
PickCanvas pc = new PickCanvas (canvas, branchGroup);  
pc.setMode (PickTool.GEOMETRY);  
pc.setShapeLocation (x0, y0);  
PickResult pr = pc.pickClosest ();  
if (pr == null) {  
    return;  
}  
Node n = pr.getNode (PickResult.TRANSFORM_GROUP);  
if (n == null) {  
    return;  
}
```



Os nós Shape3D:

```
shape.setCapability (Shape3D.ENABLE_PICK_REPORTING); // opcional  
PickTool.setCapabilities (shape, PickTool.INTERSECT_TEST);
```

Os nós TransformGroup:

```
setCapability (TransformGroup.ENABLE_PICK_REPORTING);
```