Java 多线程编程 Java 文档注释

Java Applet 基础

Applet 是一种 Java 程序。它一般运行在支持 Java 的 Web 浏览器内。因为它有完整的 Java API支持,所以Applet 是一个全功能的 Java 应用程序。

如下所示是独立的 Java 应用程序和 applet 程序之间重要的不同：

- Java 中 Applet 类继承了 java.applet.Applet 类。
- Applet 类没有定义 main()，所以一个 Applet 程序不会调用 main() 方法。
- Applet 被设计为嵌入在一个 HTML 页面。
- 当用户浏览包含 Applet 的 HTML 页面，Applet 的代码就被下载到用户的机器上。
- 要查看一个 Applet 需要 JVM。 JVM 可以是 Web 浏览器的一个插件，或一个独立的运行时环境。
- 用户机器上的 JVM 创建一个 Applet 类的实例，并调用 Applet 生命周期过程中的各种方法。
- Applet 有 Web 浏览器强制执行的严格的安全规则，Applet 的安全机制被称为沙箱安全。
- Applet 需要的其他类可以用 Java 归档（JAR）文件的形式下载下来。

Applet的生命周期

Applet 类中的四个方法给我们提供了一个框架，你可以在该框架上开发小程序：

- **init:** 该方法的目的是为你的 Applet 提供所需的任何初始化。在 Applet 标记内的 param 标签被处理后调用该方法。
- **start:** 浏览器调用 init 方法后，该方法被自动调用。每当用户从其他页面返回到包含 Applet 的页面时，则调用该方法。
- **stop:** 当用户从包含 Applet 的页面移除的时候，该方法自动被调用。因此，可以在相同的 Applet 中反复调用该方法。
- **destroy:** 此方法仅当浏览器正常关闭时调用。因为 Applet 只有在 HTML 网页上有效，所以你不应该在用户离开包含 Applet 的页面后遗漏任何资源。
- **paint:** 该方法在 start() 方法之后立即被调用，或者在 Applet 需要重绘在浏览器的时候调用。paint() 方法实际上继承于 java.awt。

"Hello, World" Applet:

下面是一个简单的 Applet 程序 HelloWorldApplet.java:

HelloWorldApplet.java 文件代码：

```
import java.applet.*;
import java.awt.*;
public class HelloWorldApplet extends Applet
{
public void paint (Graphics g)
{
g.drawString ("Hello World", 25, 50);
```

```
}
}
```

这些 import 语句将以下类导入到我们的 Applet 类中：

```
java.applet.Applet.
java.awt.Graphics.
```

没有这些 import 语句，Java 编译器就识别不了 Applet 和 Graphics 类。

Applet 类

每一个 Applet 都是 java.applet.Applet 类的子类，基础的 Applet 类提供了供衍生类调用的方法,以此来得到浏览器上下文的信息和服务。

这些方法做了如下事情：

- 得到 Applet 的参数
- 得到包含 Applet 的 HTML 文件的网络位置
- 得到 Applet 类目录的网络位置
- 打印浏览器的状态信息
- 获取一张图片
- 获取一个音频片段
- 播放一个音频片段
- 调整此 Applet 的大小

除此之外，Applet 类还提供了一个接口，该接口供 Viewer 或浏览器来获取 Applet 的信息，并且来控制 Applet 的执行。

Viewer 可能是：

- 请求 Applet 作者、版本和版权的信息
- 请求 Applet 识别的参数的描述
- 初始化 Applet
- 销毁 Applet
- 开始执行 Applet
- 结束执行 Applet

Applet 类提供了对这些方法的默认实现，这些方法可以在需要的时候重写。

"Hello，World"applet 都是按标准编写的。唯一被重写的方法是 paint 方法。

Applet 的调用

Applet 是一种 Java 程序。它一般运行在支持 Java 的 Web 浏览器内。因为它有完整的 Java API 支持,所以 Applet 是一个全功能的 Java 应用程序。

<applet> 标签是在HTML文件中嵌入 Applet 的基础。以下是一个调用"Hello World"applet的例子；

HTML 代码：

```
<html>
<title>The Hello, World Applet</title>
<hr>
<applet code="HelloWorldApplet.class" width="320" height="120">
If your browser was Java-enabled, a "Hello, World"
message would appear here.
</applet>
<hr>
</html>
```

注意: 你可以参照 HTML Applet 标签来更多的了解从 HTML 中调用 applet 的方法。

<applet> 标签的属性指定了要运行的 Applet 类。width 和 height 用来指定 Applet 运行面板的初始大小。Applet 必须使用 </applet> 标签来关闭。

如果 Applet 接受参数，那么参数的值需要在 <param> 标签里添加，该标签位于 <applet> 和 </applet> 之间。浏览器忽略了 applet 标签之间的文本和其他标签。

不支持 Java 的浏览器不能执行 <applet> 和 </applet>。因此，在标签之间显示并且和 applet 没有关系的任何东西，在不支持的 Java 的浏览器里是可见的。

Viewer 或者浏览器在文档的位置寻找编译过的 Java 代码，要指定文档的路径，得使用 <applet> 标签的 codebase 属性指定。如下所示：

```
<applet codebase="http://amrood.com/applets"
code="HelloWorldApplet.class" width="320" height="120">
```

如果 Applet 所在一个包中而不是默认包，那么所在的包必须在 code 属性里指定，例如：

```
<applet code="mypackage.subpackage.TestApplet.class"
           width="320" height="120">
```

获得applet参数

下面的例子演示了如何使用一个 Applet 响应来设置文件中指定的参数。该 Applet 显示了一个黑色棋盘图案和第二种颜色。第二种颜色和每一列的大小通过文档中的 Applet 的参数指定。

CheckerApplet 在 init() 方法里得到它的参数。也可以在 paint() 方法里得到它的参数。然而，在 Applet 开始得到值并保存了设置，而不是每一次刷新的时候都得到值，这样是很方便，并且高效的。

Applet viewer 或者浏览器在 Applet 每次运行的时候调用 init() 方法。在加载 Applet 之后，Viewer 立即调用 init() 方法（Applet.init()什么也没做），重写该方法的默认实现，添加一些自定义的初始化代码。

Applet.getParameter() 方法通过给出参数名称得到参数值。如果得到的值是数字或者其他非字符数据，那么必须解析为字符串类型。

下例是 CheckerApplet.java 的修改：

CheckerApplet.java 文件代码：

```
import java.applet.*;
import java.awt.*;
public class CheckerApplet extends Applet
{
int squareSize = 50;// 初始化默认大小
public void init () {}
private void parseSquareSize (String param) {}
private Color parseColor (String param) {}
public void paint (Graphics g) {}
}
```

下面是 CheckerApplet 类的 init() 方法和私有的 parseSquareSize() 方法：

```
public void init ()
{
String squareSizeParam = getParameter ("squareSize");
parseSquareSize (squareSizeParam);
String colorParam = getParameter ("color");
Color fg = parseColor (colorParam);
setBackground (Color.black);
setForeground (fg);
}
private void parseSquareSize (String param)
{
if (param == null) return;
try {
squareSize = Integer.parseInt (param);
}
catch (Exception e) {
// 保留默认值
}
}
```

该 Applet 调用 parseSquareSize()，来解析 squareSize 参数。parseSquareSize() 调用了库方法 Integer. parseInt() 该方法将一个字符串解析为一个整数，当参数无效的时候，Integer.parseInt() 抛出异常。

因此，parseSquareSize() 方法也是捕获异常的，并不允许 Applet 接受无效的输入。

Applet 调用 parseColor()方法将颜色参数解析为一个 Color 值。parseColor() 方法做了一系列字符串的比较，来匹配参数的值和预定义颜色的名字。你需要实现这些方法来使 Applet 工作。

指定 applet 参数

如下的例子是一个HTML文件，其中嵌入了 CheckerApplet 类。HTML文件通过使用 <param> 标签的方法给 applet 指定了两个参数。

```
<html>
<title>Checkerboard Applet</title>
<hr>
<applet code="CheckerApplet.class" width="480" height="320">
<param name="color" value="blue">
<param name="squaresize" value="30">
</applet>
<hr>
</html>
```

注意: 参数名字大小写不敏感。

应用程序转换成 Applet

将图形化的 Java 应用程序（是指，使用AWT的应用程序和使用 java 程序启动器启动的程序）转换成嵌入在web页面里的applet是很简单的。

下面是将应用程序转换成 Applet 的几个步骤：

- 编写一个 HTML 页面，该页面带有能加载 applet 代码的标签。
- 编写一个 JApplet 类的子类，将该类设置为 public。否则，Applet 不能被加载。
- 消除应用程序的 main()方法。不要为应用程序构造框架窗口，因为你的应用程序要显示在浏览器中。
- 将应用程序中框架窗口的构造方法里的初始化代码移到 Applet 的 init() 方法中，你不必显示的构造 Applet 对象，浏览器将通过调用 init() 方法来实例化一个对象。
- 移除对 setSize() 方法的调用，对于 Applet 来讲，大小已经通过 HTML 文件里的 width 和 height 参数设定好了。
- 移除对 setDefaultCloseOperation() 方法的调用。Applet 不能被关闭，它随着浏览器的退出而终止。
- 如果应用程序调用了 setTitle() 方法，消除对该方法的调用。applet 不能有标题栏。（当然你可以给通过 html 的 title 标签给网页自身命名）
- 不要调用 setVisible(true),Applet 是自动显示的。

事件处理

Applet 类从 Container 类继承了许多事件处理方法。Container 类定义了几个方法，例如：processKeyEvent() 和processMouseEvent()，用来处理特别类型的事件，还有一个捕获所有事件的方法叫做 processEvent。

为了响应一个事件，Applet 必须重写合适的事件处理方法。

ExampleEventHandling.java 文件代码：

```
import java.awt.event.MouseListener;
import java.awt.event.MouseEvent;
import java.applet.Applet;
import java.awt.Graphics;
public class ExampleEventHandling extends Applet
implements MouseListener {
StringBuffer strBuffer;
public void init() {
addMouseListener(this);
```

```
strBuffer = new StringBuffer();
addItem("initializing the applet ");
}
public void start() {
addItem("starting the applet ");
}
public void stop() {
addItem("stopping the applet ");
}
public void destroy() {
addItem("unloading the applet");
}
void addItem(String word) {
System.out.println(word);
strBuffer.append(word);
repaint();
}
public void paint(Graphics g) {
//Draw a Rectangle around the applet's display area.
g.drawRect(0, 0,
getWidth() - 1,
getHeight() - 1);
//display the string inside the rectangle.
g.drawString(strBuffer.toString(), 10, 20);
}
public void mouseEntered(MouseEvent event) {
}
public void mouseExited(MouseEvent event) {
}
public void mousePressed(MouseEvent event) {
}
public void mouseReleased(MouseEvent event) {
}
public void mouseClicked(MouseEvent event) {
addItem("mouse clicked! ");
}
}
```

如下调用该 Applet：

```
<html>
<title>Event Handling</title>
<hr>
<applet code="ExampleEventHandling.class"
width="300" height="300">
</applet>
<hr>
</html>
```

最开始运行，Applet 显示 "initializing the applet. Starting the applet."，然后你一点击矩形框，就会显示 "mouse clicked" 。

显示图片

Applet 能显示 GIF,JPEG,BMP 等其他格式的图片。为了在 Applet 中显示图片，你需要使用 java.awt.Graphics 类的drawImage ()方法。

如下实例演示了显示图片的所有步骤：

ImageDemo.java 文件代码：

```
import java.applet.*;
import java.awt.*;
import java.net.*;
public class ImageDemo extends Applet
{
private Image image;
private AppletContext context;
public void init()
{
context = this.getAppletContext();
String imageURL = this.getParameter("image");
if(imageURL == null)
{
imageURL = "java.jpg";
}
try
{
URL url = new URL(this.getDocumentBase(), imageURL);
image = context.getImage(url);
}catch(MalformedURLException e)
{
e.printStackTrace();
// Display in browser status bar
context.showStatus("Could not load image!");
}
}
public void paint(Graphics g)
{
context.showStatus("Displaying image");
g.drawImage(image, 0, 0, 200, 84, null);
g.drawString("www.javalicense.com", 35, 100);
}
}
```

如下调用该applet：

```
<html>
<title>The ImageDemo applet</title>
<hr>
<applet code="ImageDemo.class" width="300" height="200">
<param name="image" value="java.jpg">
</applet>
<hr>
</html>
```

播放音频

Applet 能通过使用 java.applet 包中的 AudioClip 接口播放音频。AudioClip 接口定义了三个方法：

- **public void play():** 从一开始播放音频片段一次。
- **public void loop():** 循环播放音频片段
- **public void stop():** 停止播放音频片段

为了得到 AudioClip 对象，你必须调用 Applet 类的 getAudioClip() 方法。无论 URL 指向的是否是一个真实的音频文件，该方法都会立即返回结果。

直到要播放音频文件时，该文件才会下载下来。

如下实例演示了播放音频的所有步骤：

AudioDemo.java 文件代码：

```
import java.applet.*;
import java.awt.*;
import java.net.*;
public class AudioDemo extends Applet
{
private AudioClip clip;
private AppletContext context;
public void init()
{
context = this.getAppletContext();
String audioURL = this.getParameter("audio");
if(audioURL == null)
{
audioURL = "default.au";
}
try
{
URL url = new URL(this.getDocumentBase(), audioURL);
clip = context.getAudioClip(url);
}catch(MalformedURLException e)
{
e.printStackTrace();
context.showStatus("Could not load audio file!");
}
}
public void start()
{
if(clip != null)
{
clip.loop();
}
}
public void stop()
{
if(clip != null)
{
clip.stop();
}
```

```
}
}
```

如下调用applet：

```
<html>
<title>The ImageDemo applet</title>
<hr>
<applet code="ImageDemo.class" width="0" height="0">
<param name="audio" value="test.wav">
</applet>
<hr>
```

你可以使用你电脑上的 test.wav 来测试上面的实例。

Java 多线程编程 Java 文档注释