
Class java.awt.MediaTracker

```
java.lang.Object
   |
   +----java.awt.MediaTracker
```

---

public class **MediaTracker**
extends Object

A utility class to track the status of a number of media objects. Media objects could include images as well as audio clips, though currently only images are supported. To use it, simply create an instance and then call addImage() for each image to be tracked. Each image can be assigned a unique ID for indentification purposes. The IDs control the priority order in which the images are fetched as well as identifying unique subsets of the images that can be waited on independently. Here is an example:

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Image;
import java.awt.Graphics;
import java.awt.MediaTracker;
public class ImageBlaster extends Applet implements Runnable {
        MediaTracker tracker;
        Image bg;
        Image anim[] = new Image[5];
        int index;
        Thread animator;
        // Get the images for the background (id == 0) and the animation
        // frames (id == 1) and add them to the MediaTracker
        public void init() {
            tracker = new MediaTracker(this);
            bg = getImage(getDocumentBase(), "images/background.gif");
            tracker.addImage(bg, 0);
            for (int i = 0; i < 5; i++) {
                anim[i] = getImage(getDocumentBase(), "images/anim"+i+".gif");
                tracker.addImage(anim[i], 1);
            }
        }
        // Start the animation thread.
        public void start() {
            animator = new Thread(this);
            animator.start();
        }
        // Stop the animation thread.
        public void stop() {
            animator.stop();
            animator = null;
        }
        // Run the animation thread.
        // First wait for the background image to fully load and paint.
        // Then wait for all of the animation frames to finish loading.
        // Finally loop and increment the animation frame index.
        public void run() {
            try {
                tracker.waitForID(0);
                tracker.waitForID(1);
            } catch (InterruptedException e) {
                return;
            }
            Thread me = Thread.currentThread();
            while (animator == me) {
                try {
                    Thread.sleep(100);
                } catch (InterruptedException e) {
                    break;
                }
                synchronized (this) {
                    index++;
                    if (index >= anim.length) {
                        index = 0;
                    }
                }
                repaint();
            }
        }
        // The background image fills our frame so we don't need to clear
        // the applet on repaints, just call the paint method.
        public void update(Graphics g) {
            paint(g);
        }
        // Paint a large red rectangle if there are any errors loading the
        // images.  Otherwise always paint the background so that it appears
        // incrementally as it is loading.  Finally, only paint the current
        // animation frame if all of the frames (id == 1) are done loading
        // so that we don't get partial animations.
        public void paint(Graphics g) {
            if (tracker.isErrorAny()) {
                g.setColor(Color.red);
                g.fillRect(0, 0, size().width, size().height);
                return;
            }
            g.drawImage(bg, 0, 0, this);
            if (tracker.checkID(1)) {
                g.drawImage(anim[index], 10, 10, this);
            }
        }
}
```

Version:

1.6, 10/08/95

Author:

Jim Graham

Constructor Index

o **MediaTracker**(Component)
  Create a Media tracker to track images for a given Component.

Method Index

o **addImage**(Image, int)
  Add an image to the list of images being tracked.
o **addImage**(Image, int, int, int)
  Add a scaled image to the list of images being tracked.
o **checkAll**()
  Check to see if all images have finished loading, but do not start loading the images if they are not already loading.
o **checkAll**(boolean)
  Check to see if all images have finished loading and start loading any images that are not yet being loaded if load is true.
o **checkID**(int)
  Check to see if all images tagged with the indicated ID have finished loading, but do not start loading the images if they are not already loading.
o **checkID**(int, boolean)
  Check to see if all images tagged with the indicated ID have finished loading and start loading any images with that ID that are not yet being loaded if load is true.
o **isErrorAny**()
  Check the error status of all of the images.
o **isErrorID**(int)
  Check the error status of all of the images with the specified ID.
o **waitForAll**()
  Start loading all images and wait until they have finished loading or receive an error.
o **waitForID**(int)
  Start loading all images with the specified ID and wait until they have finished loading or receive an error.

Constructors

o **MediaTracker**

```
 public MediaTracker(Component comp)
```

Create a Media tracker to track images for a given Component.
Parameters:
comp – the component on which the images will eventually be drawn

Methods

o addImage

```
public void addImage(Image image,
                     int id)
```

Add an image to the list of images being tracked. The image will eventually be rendered at its default (unscaled) size.

Parameters:

image – the image to be tracked
id – the identifier used to later track this image

o addImage

```
public synchronized void addImage(Image image,
                                  int id,
                                  int w,
                                  int h)
```

Add a scaled image to the list of images being tracked. The image will eventually be rendered at the indicated size.

Parameters:

image – the image to be tracked
id – the identifier used to later track this image
w – the width that the image will be rendered at
h – the height that the image will be rendered at

o checkAll

```
public boolean checkAll()
```

Check to see if all images have finished loading, but do not start loading the images if they are not already loading. If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.

Returns:

true if all images have finished loading or have an error

See Also:

checkAll, checkID, isErrorAny, isErrorID

o checkAll

```
public synchronized boolean checkAll(boolean load)
```

Check to see if all images have finished loading and start loading any images that are not yet being loaded if load is true. If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.

Parameters:
load – start loading the images if this parameter is true
Returns:
true if all images have finished loading or have an error
See Also:
isErrorAny, isErrorID, checkID, checkAll

o **isErrorAny**

```
public synchronized boolean isErrorAny()
```

Check the error status of all of the images.
Returns:
true if any of the images had an error during loading
See Also:
isErrorID

o **waitForAll**

```
public synchronized void waitForAll() throws InterruptedException
```

Start loading all images and wait until they have finished loading or receive an error. If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.
See Also:
waitForID, isErrorAny, isErrorID

o **checkID**

```
public boolean checkID(int id)
```

Check to see if all images tagged with the indicated ID have finished loading, but do not start loading the images if they are not already loading. If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.
Parameters:
id – the identifier used to determine which images to check
Returns:
true if all tagged images have finished loading or have an error
See Also:
checkID, checkAll, isErrorAny, isErrorID

o **checkID**

```
public synchronized boolean checkID(int id,
                                    boolean load)
```

Check to see if all images tagged with the indicated ID have finished loading and start loading any images with that ID that are not yet being loaded if load is true.

If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.

Parameters:

id – the identifier used to determine which images to check

load – start loading the images if this parameter is true

Returns:

true if all tagged images have finished loading or have an error

See Also:

checkID, checkAll, isErrorAny, isErrorID

o **isErrorID**

```
public synchronized boolean isErrorID(int id)
```

Check the error status of all of the images with the specified ID.

Parameters:

id – the identifier used to determine which images to check

Returns:

true if any of the tagged images had an error during loading

See Also:

isErrorAny

o **waitForID**

```
public synchronized void waitForID(int id) throws InterruptedException
```

Start loading all images with the specified ID and wait until they have finished loading or receive an error. If there is an error while loading or scaling an image then that image is considered "complete". Use isErrorAny or isErrorID to check for errors.

See Also:

waitForAll, isErrorAny, isErrorID

All Packages   This Package   Previous   Next