Прежде чем начать использовать JOGL, необходимо скачать и инсталлировать файлы JOGL JAR и нативные JARs или нативные библиотечные файлы. Последняя стабильная версия JOGL может быть получена по адресу <https://jogamp.org/deployment/jogamp-current/archive/> .

При переходе появится страница, показанная на рис. 1.

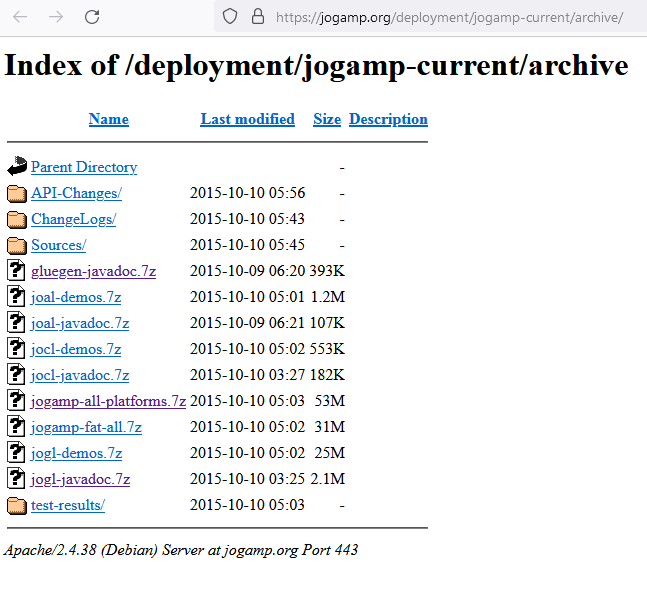


Рисунок 1. Страница загрузки стабильной версии JOGL

Скачайте и распакуйте в каком-либо каталоге файл [jogamp-all-platforms.7z](https://jogamp.org/deployment/jogamp-current/archive/jogamp-all-platforms.7z). Содержимое этого каталога показано на рис. 2.

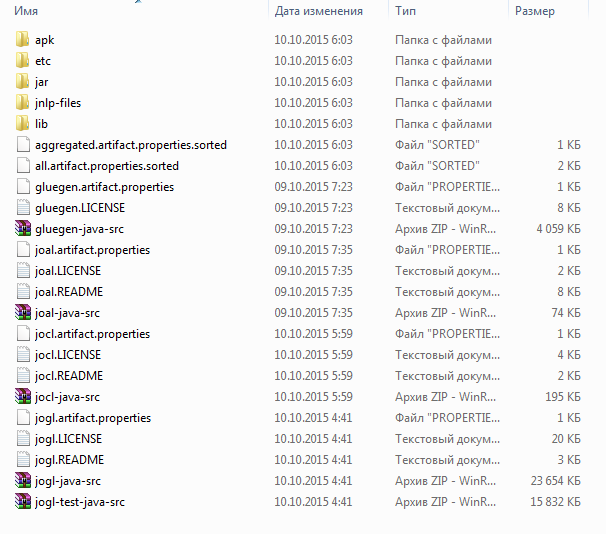


Рисунок 2. Содержимое каталога после распаковки файла jogamp-all-platforms.7z.

Перейдите в каталог jar. Часть его содержимого показана на рис. 3.

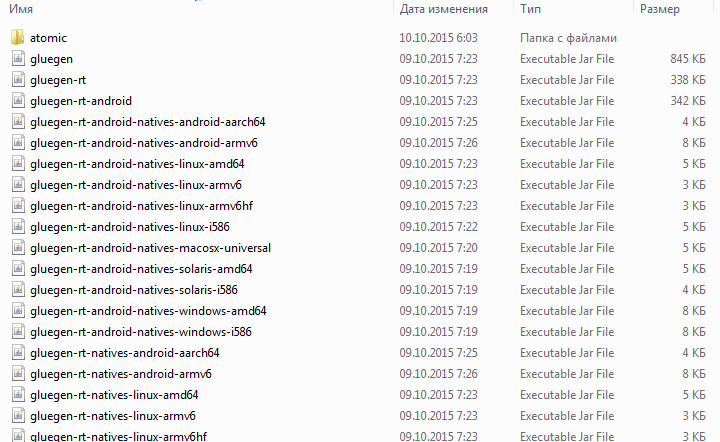


Рисунок 3. Часть содержимого каталога jar

Для каждой платформы необходимы определенные файлы.

|  |  |  |
| --- | --- | --- |
| **64-bit Windows** | **64-bit Linux** | **32/64-bit Mac** |
| gluegen-rt.jar | gluegen-rt.jar | gluegen-rt.jar |
| jogl-all.jar | jogl-all.jar | jogl-all.jar |
| gluegen-java-src.zip | gluegen-java-src.zip | gluegen-java-src.zip |
| jogl-java-src.zip | jogl-java-src.zip | jogl-java-src.zip |
| gluegen-rt-natives-windows-amd64.jar | gluegen-rt-natives-linux-amd64.jar | gluegen-rt-natives-macosx-universal.jar |
| jogl-all-natives-windows-amd64.jar | jogl-all-natives-linux-amd64.jar | jogl-all-natives-macosx-universal.jar |

|  |  |
| --- | --- |
| **32-bit Windows** | **32-bit Linux** |
| gluegen-rt.jar | gluegen-rt.jar |
| jogl-all.jar | jogl-all.jar |  |
| gluegen-java-src.zip | gluegen-java-src.zip |  |
| jogl-java-src.zip | jogl-java-src.zip |  |
| gluegen-rt-natives-windows-i586.jar | gluegen-rt-natives-linux-i586.jar |  |
| jogl-all-natives-windows-i586.jar | jogl-all-natives-linux-i586.jar |  |

**Создание проекта JOGL в Eclipse**

Самый простой путь создания проекта с использованием JOGL состоит в подключении необходимых jar-файлов к проекту с использованием в дальнейшем зависимости от этого проекта.

**Создание проекта JOGL**

* Поместите все JAR с кодом, нативные JARs для всех платформ, которые должно поддерживать ваше приложение, и ZIP – файлы с исходными кодами в каталог JOGL ***в каталоге вашего рабочего пространства Eclipse***. Если это платформа **Windows 64-bit**, то выберите следующие файлы:

- gluegen-rt.jar,

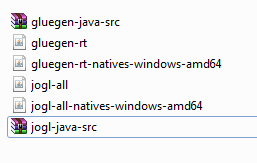
- jogl-all.jar,

- gluegen-java-src.zip,

- jogl-java-src.zip,

- gluegen-rt-natives-windows-amd64.jar,

- jogl-all-natives-windows-amd64.jar.



* В Eclipse выберите File > New > Java Project. Укажите имя проекта «JOGL» и нажмите кнопку «Next» (рис. 4).

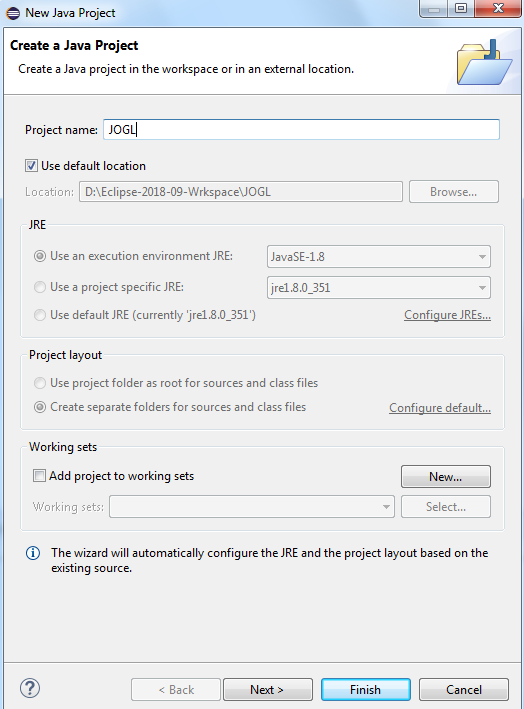


Рисунок 4. Создание проекта с именем «JOGL»

* Удалите все нативные jar из проекта, оставив только gluegen-rt.jar и jogl-all.jar (рис. 5).

Файлы gluegen-rt-natives-\*-\*.jar и jogl-all-natives-\*-\*.jar не требуются в пути к классам JOGL, но требуются Eclipse для экспорта проекта в виде файла JAR, который можно запускать.

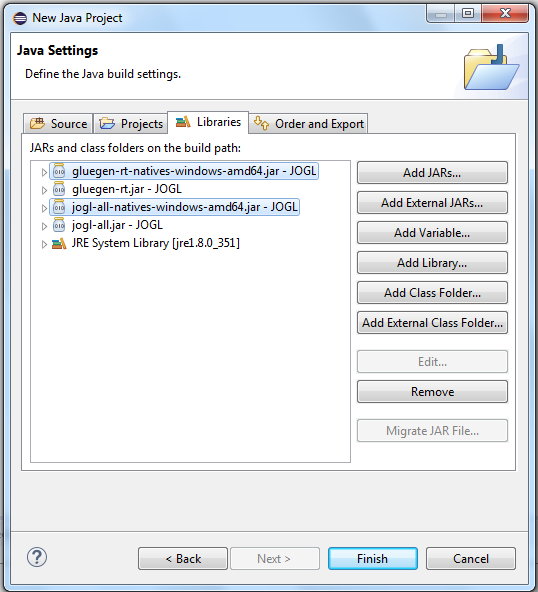


Рисунок 5. Удаление нативных JAR

* Разверните JAR – файлы jogl-all и gluegen-rt. Выберите "Source attachment" (прикрепить источник), укажите соответствующий файл с именем \*java-src.zip в проекте (рис. 6).

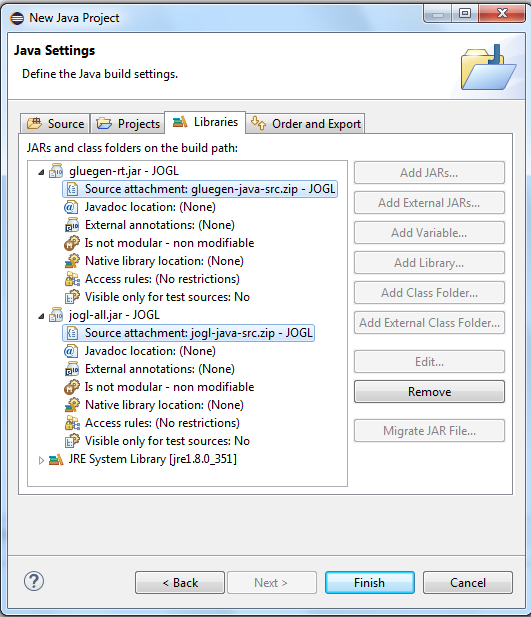


Рисунок 6. Прикрепление файлов с исходным кодом

Выберите закладку "Order and Export". Выберите оба JAR и нажмите кнопку "Finish" (рис. 7).

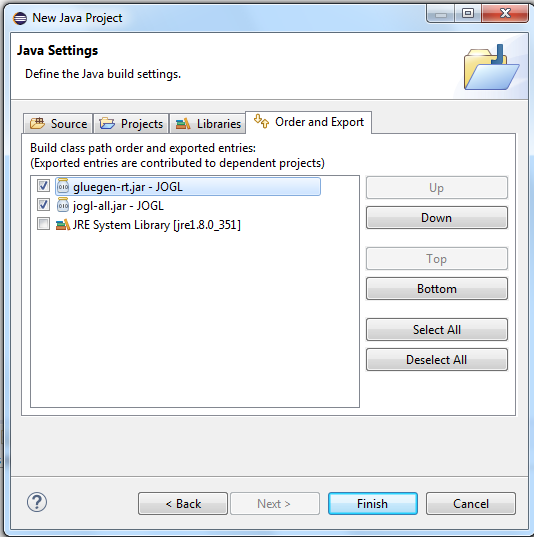


Рисунок 7. Экспорт JAR - файлов JOGL

В Eclipse получим следующую структуру проекта с именем «JOGL» (рис. 8).

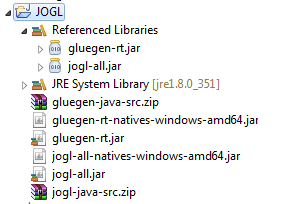


Рисунок 8. Структура проекта с именем «JOGL»

**Добавление зависимости от проекта JOGL**

* Создайте в Eclipse новый проект, например, NewJOGLProject.
* Нажмите правую кнопку мыши на новом проекте NewJOGLProject и выберите "Properties" (в самом низу выпадающего списка).
* Выберите в левом столбце появившегося окна пункт "Java Build Path".
* Нажмите на кнопку "Add...", выберите проект "JOGL" и нажмите "OK" (рис. 9).

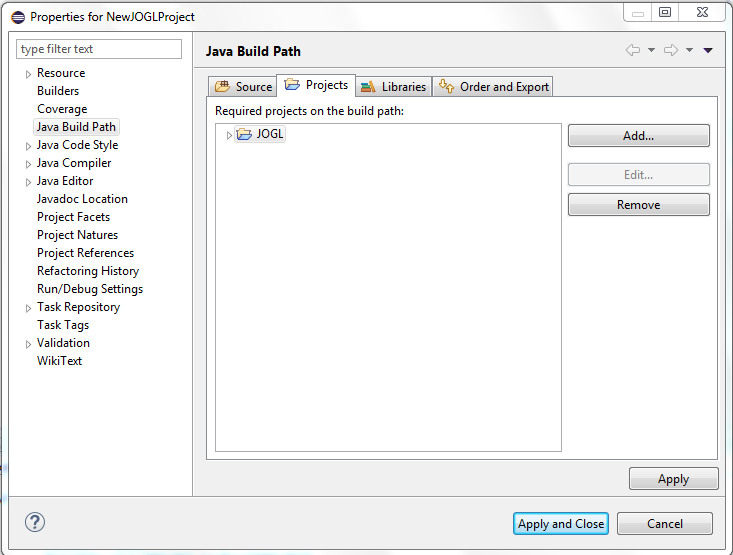


Рисунок 9. Добавление зависимости проекта NewJOGLProject от проекта с именем «JOGL»

Теперь в проекте NewJOGLProject можно использовать библиотеку JOGL. Если это необходимо, то можно создать и другие проекты, зависимые от проекта с именем «JOGL».

**IntelliJ IDEA**

Чтобы начать работать с OpenGL нужно сначала скачать архив jogamp-all-platforms.7z с сайта:

<https://jogamp.org/wiki/index.php/Downloading_and_installing_JOGL>

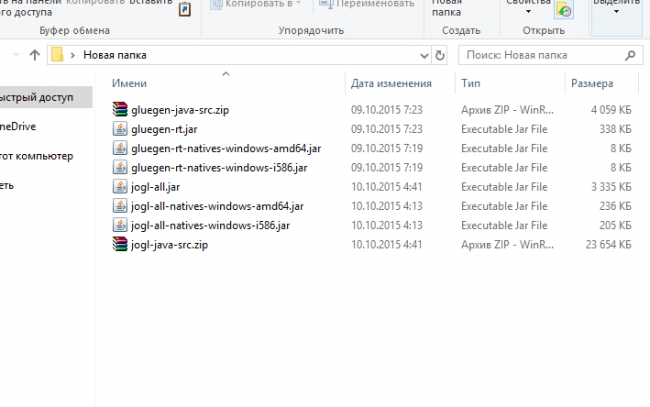
После того как архив получен с сайта, извлекаем в новую папку следующие файлы:

Файлы zip:

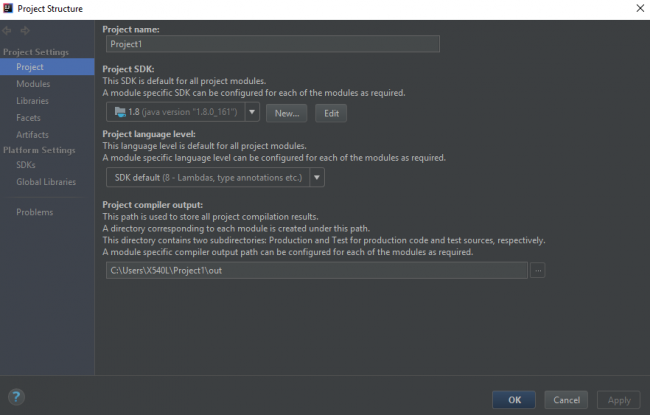
* jogl-java-src.zip
* gluegen-java-src.zip

Из папки jar:

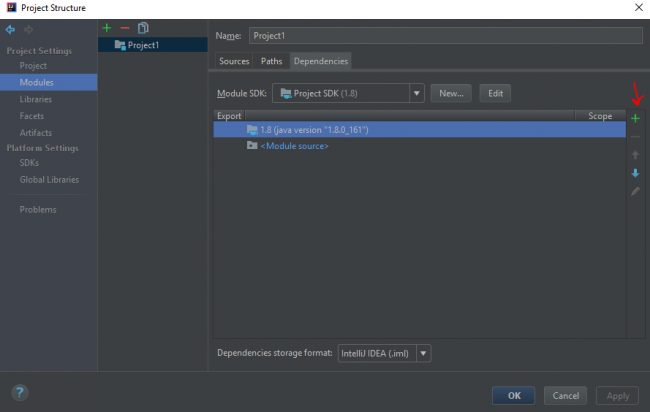
* jogl-all.jar
* jogl-all-natives-windows-amd64.jar
* gluegen-rt.jar
* gluegen-rt-natives-windows-amd64.jar



Запускаем IntelliJ IDEA и создаем проект или открываем ранее созданный. Заходим в меню File и выбираем пункт Project Structure.

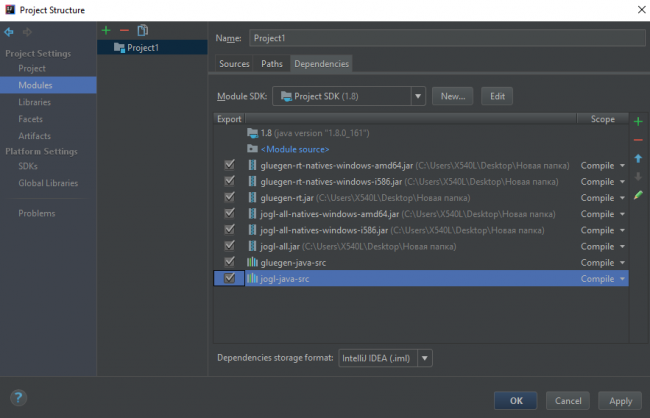


Переходим в **Modules** и нажимаем на зеленый плюс (может быть и не зеленый и находиться в другом месте, это зависит от версии IntelliJ IDEA).



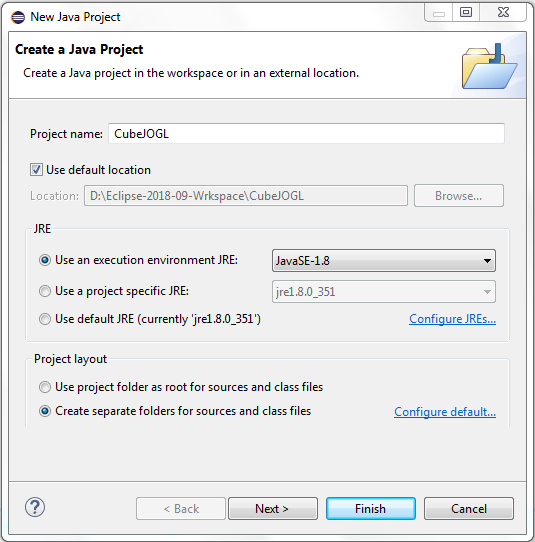
Далее выбираем JAR из папки, в которую извлекли файлы, добавляем файлы с разрешением jar. Затем выбираем Library, а потом Java и добавляем файлы с разрешением zip. Отмечаем все что добавили галочками и нажимаем Apply и затем OK.

Теперь JOGL подключен. Если нажать в проекте на **External Libraries**, то можно увидеть все что мы подключили.

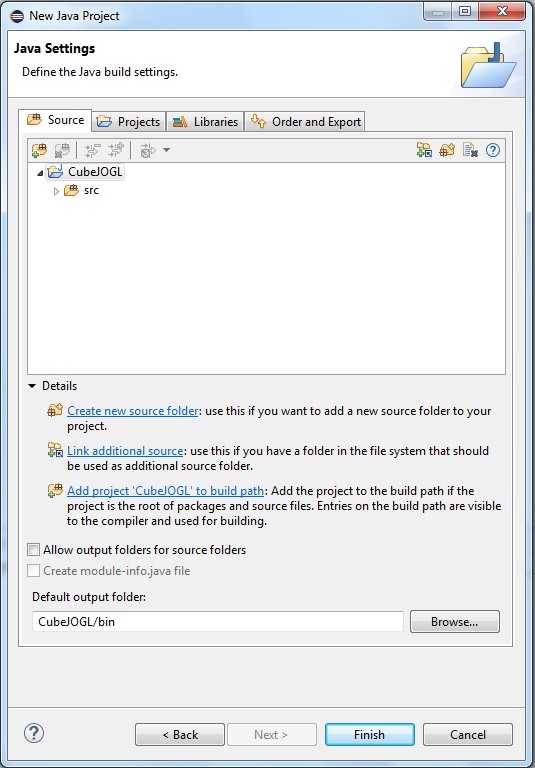


**Пробный проект JOGL в Eclipse**

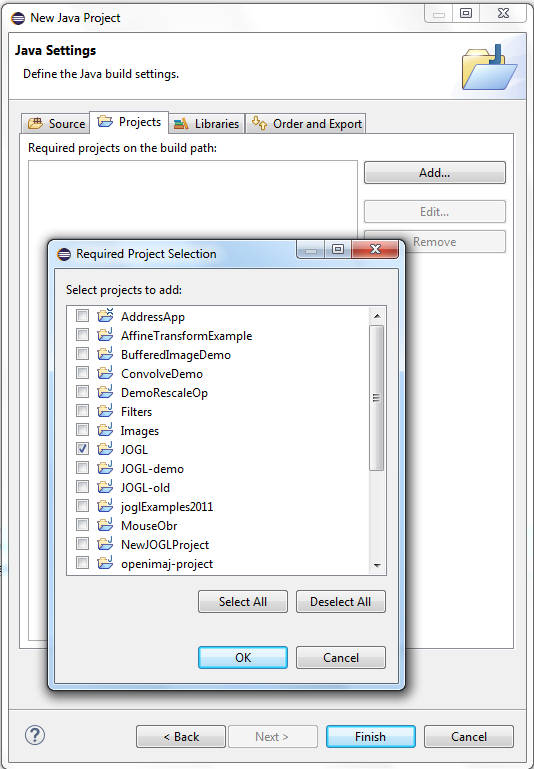
1. Создайте в Eclipse Java – проект CubeJOGL.



1. Нажмите кнопку Next. ***Нужно нажать именно Next, а не Finish***. Это нужно для того, чтобы определить зависимость от проекта JOGL, который мы создали ранее. Появится следующее окно:



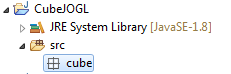
1. Выберите закладку Projects и нажмите кнопку Add (расположена справа в закладке Projects). Появится окно выбора проектов. Выберите проект JOGL.



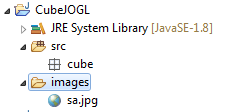
1. Нажмите кнопку Ok. Затем нажмите кнопку Finish в окне создания проекта.
2. Создастся проект со следующей структурой:



1. В проекте CubeJOGL создайте пакет cube.



1. В проекте CubeJOGL создайте каталог images и скопируйте в него файл sa.jpg (находится на сетевом диске).



1. Создайте в пакете *cube* класс *CubeTexture*. Код, который необходимо поместить в класс *CubeTexture*, приводится ниже.

**package** cube;

**import** java.awt.DisplayMode;

**import** java.io.File;

**import** java.io.IOException;

**import** javax.swing.JFrame;

**import** com.jogamp.opengl.GL2;

**import** com.jogamp.opengl.GLAutoDrawable;

**import** com.jogamp.opengl.GLCapabilities;

**import** com.jogamp.opengl.GLEventListener;

**import** com.jogamp.opengl.GLProfile;

**import** com.jogamp.opengl.awt.GLCanvas;

**import** com.jogamp.opengl.glu.GLU;

**import** com.jogamp.opengl.util.FPSAnimator;

**import** com.jogamp.opengl.util.texture.Texture;

**import** com.jogamp.opengl.util.texture.TextureIO;

**public** **class** CubeTexture **implements** GLEventListener {

**public** **static** DisplayMode *dm*, *dm\_old*;

**private** GLU glu = **new** GLU();

**private** **float** xrot,yrot,zrot;

**private** **int** texture;

@Override

**public** **void** display(GLAutoDrawable drawable) {

**final** GL2 gl = drawable.getGL().getGL2();

gl.glClear(GL2.***GL\_COLOR\_BUFFER\_BIT*** | GL2.***GL\_DEPTH\_BUFFER\_BIT***);

gl.glLoadIdentity();

gl.glTranslatef(0f, 0f, -5.0f);

gl.glRotatef(xrot, 1.0f, 0.0f, 0.0f);

gl.glRotatef(yrot, 0.0f, 1.0f, 0.0f);

gl.glRotatef(zrot, 0.0f, 0.0f, 1.0f);

gl.glBindTexture(GL2.***GL\_TEXTURE\_2D***, texture);

gl.glBegin(GL2.***GL\_QUADS***);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f(-1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f( 1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f( 1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f(-1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f(-1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f(-1.0f, 1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f( 1.0f, 1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f( 1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f(-1.0f, 1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f(-1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f( 1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f( 1.0f, 1.0f, -1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f(-1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f( 1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f( 1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f(-1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f( 1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f( 1.0f, 1.0f, -1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f( 1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f( 1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(0.0f, 0.0f); gl.glVertex3f(-1.0f, -1.0f, -1.0f);

gl.glTexCoord2f(1.0f, 0.0f); gl.glVertex3f(-1.0f, -1.0f, 1.0f);

gl.glTexCoord2f(1.0f, 1.0f); gl.glVertex3f(-1.0f, 1.0f, 1.0f);

gl.glTexCoord2f(0.0f, 1.0f); gl.glVertex3f(-1.0f, 1.0f, -1.0f);

gl.glEnd();

gl.glFlush();

xrot += .1f;

yrot += .1f;

zrot += .1f;

}

@Override

**public** **void** dispose(GLAutoDrawable drawable) {

}

@Override

**public** **void** init(GLAutoDrawable drawable) {

**final** GL2 gl = drawable.getGL().getGL2();

gl.glShadeModel(GL2.***GL\_SMOOTH***);

gl.glClearColor(0f, 0f, 0f, 0f);

gl.glClearDepth(1.0f);

gl.glEnable(GL2.***GL\_DEPTH\_TEST***);

gl.glDepthFunc(GL2.***GL\_LEQUAL***);

gl.glHint(GL2.***GL\_PERSPECTIVE\_CORRECTION\_HINT***, GL2.***GL\_NICEST***);

gl.glEnable(GL2.***GL\_TEXTURE\_2D***);

**try**{

File im = **new** File("images\\sa.jpg");

Texture t = TextureIO.*newTexture*(im, **true**);

texture= t.getTextureObject(gl);

}**catch**(IOException e){

e.printStackTrace();

}

}

@Override

**public** **void** reshape(GLAutoDrawable drawable, **int** x, **int** y, **int** width, **int** height) {

**final** GL2 gl = drawable.getGL().getGL2();

**if**(height <= 1)

height = 1;

**final** **float** h = (**float**) width / (**float**) height;

gl.glViewport(0, 0, width, height);

gl.glMatrixMode(GL2.***GL\_PROJECTION***);

gl.glLoadIdentity();

glu.gluPerspective(45.0f, h, 1.0, 20.0);

gl.glMatrixMode(GL2.***GL\_MODELVIEW***);

gl.glLoadIdentity();

}

**public** **static** **void** main(String[] args) {

**final** GLProfile profile = GLProfile.*get*(GLProfile.***GL2***);

GLCapabilities capabilities = **new** GLCapabilities(profile);

**final** GLCanvas glcanvas = **new** GLCanvas(capabilities);

CubeTexture r = **new** CubeTexture();

glcanvas.addGLEventListener(r);

glcanvas.setSize(400, 400);

**final** JFrame frame = **new** JFrame (" Куб с текстурой ");

frame.getContentPane().add(glcanvas);

frame.setSize(frame.getContentPane().getPreferredSize());

frame.setVisible(**true**);

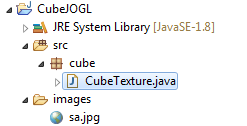
**final** FPSAnimator animator = **new** FPSAnimator(glcanvas, 300, **true**);

animator.start();

}

}

1. Теперь проект CubeJOGL имеет следующую структуру:



1. Запустите класс CubeJOGL на выполнение. Если вы увидели вращающейся куб, значит все сделано правильно и у вас заработала программа с использованием JOGL.

**Базовый фрейм JOGL**

Чтобы программа могла использовать API JOGL, необходимо реализовать интерфейс GLEventListener.

**Интерфейс GLEventListener**

Все методы интерфейса GLEventListenerтребуют в качестве параметра объект класса, реализующего интерфейса GLAutoDrawable. Тип возвращаемого значения всех методов - void.

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| [**display**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) drawable) | Вызывается drawable для инициализации рендеринга OpenGL клиентом. Метод содержит логику, используемую для рисования графических элементов с использованием OpenGL API. |
| [**dispose**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#dispose(com.jogamp.opengl.GLAutoDrawable))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) drawable) | Уведомляет слушателя о выполнении освобождения всех ресурсов OpenGL для каждого GLContext, таких как буферы памяти и программы GLSL. |
| [**init**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#init(com.jogamp.opengl.GLAutoDrawable))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) drawable) | Вызывается drawable сразу после инициализации контекста OpenGL. |
| [**reshape**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#reshape(com.jogamp.opengl.GLAutoDrawable,int,int,int,int))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) drawable, int x, int y, int width, int height) | Вызывается объектом рисования во время первого перерисовывания после изменения размера компонента.  **Параметры:**  drawable - запуск [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html)  x - x-координата окна просмотра в пикселях  y - y-координата окна просмотра в пикселях  width - ширина области просмотра в пикселях  height - высота окна просмотра в пикселях |

**Интерфейс GLAutoDrawable**

GLAutoDrawable создает основной контекст рендеринга JOGL.

Интерфейс GLAutoDrawable расширяет интерфейсы [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) и [NativeSurfaceHolder](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurfaceHolder.html).

Этот интерфейс реализуют классы ***GLAutoDrawableBase,*** [***GLAutoDrawableDelegate***](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawableDelegate.html)***,*** [***GLCanvas***](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html)***,*** [***GLJCanvas***](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/swt/GLCanvas.html)***,*** [***GLJPanel***](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html)***,***[***GLWindow***](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/newt/opengl/GLWindow.html).

Методы интерфейса GLAutoDrawable

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#addGLEventListener(int,com.jogamp.opengl.GLEventListener))​(int index, [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет заданный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу этой очереди рисования. |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#addGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет заданное значение [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) в конец очереди рисования. |
| boolean | [**areAllGLEventListenerInitialized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#areAllGLEventListenerInitialized())() | Возвращает true, если все [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) инициализированы, в противном случае false . |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**createContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#createContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) shareWith) | Создает новый контекст для рисования для этого объекта рисования, который при необходимости будет совместно использовать объекты буфера, текстуры и другие объекты OpenGL на стороне сервера с указанным GLContext. |
| void | [**destroy**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#destroy())() | Уничтожает все ресурсы, связанные с этим GLAutoDrawable, включая GLContext. |
| void | [**display**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())() | Вызывает выполнение рендеринга OpenGL для этого GLAutoDrawable в следующем порядке: вызов [display(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)) всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**disposeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#disposeGLEventListener(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean remove) | Удаляет указанный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) посредством [dispose(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#dispose(com.jogamp.opengl.GLAutoDrawable)), если он был инициализирован и добавлен в очередь. |
| void | [**flushGLRunnables**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#flushGLRunnables())() | Сбрасывает все [enqueued](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable)) [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) этого [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), включая уведомление ожидающего исполнителя. |
| [**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) | [**getAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getAnimator())() |  |
| boolean | [**getAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getAutoSwapBufferMode())() | Указывает, включена ли автоматическая замена буфера для этого объекта рисования. |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**getContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getContext())() | Возвращает контекст, связанный с этим объектом рисования.  Возвращенный контекст будет синхронизирован. Не полагайтесь на его идентификатор, контекст может измениться. |
| int | [**getContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getContextCreationFlags())() |  |
| [**GLDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) | [**getDelegatedDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getDelegatedDrawable())() | Если реализация использует делегирование, возвращает делегированный экземпляр [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html), в противном случае возвращает экземпляр this. |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**getExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getExclusiveContextThread())() |  |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**getGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getGL())() | Возвращает объект конвейера [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html), который использует этот GLAutoDrawable. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**getGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getGLEventListener(int))​(int index) | Возвращает значение [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу очереди для рисования. |
| int | [**getGLEventListenerCount**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getGLEventListenerCount())() | Возвращает номер [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) этой очереди для рисования. |
| boolean | [**getGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getGLEventListenerInitState(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Возвращает, инициализирован ли данный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) или нет. |
| [**RecursiveLock**](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/util/locks/RecursiveLock.html?is-external=true) | [**getUpstreamLock**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamLock())() | Возвращает рекурсивный объект блокировки [upstream widget](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamWidget()) для синхронизации многопоточного доступа поверх [NativeSurface.lockSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()). |
| [**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) | [**getUpstreamWidget**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamWidget())() | Метод *может* возвращать вышестоящий объект инструментария пользовательского интерфейса, содержащий экземпляр [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), если он существует. |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable))​(boolean wait, [**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) glRunnable) | Ставит в очередь одноразовый запрос [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html), который будет выполнен в рамках следующего вызова [display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())после вызова всех зарегистрированных методов [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)[display(GLAutoDrawable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)). |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,java.util.List))​(boolean wait, [**List**](http://docs.oracle.com/javase/6/docs/api/java/util/List.html?is-external=true)<[**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html)> glRunnables) | Расширяет [invoke(boolean, GLRunnable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable))функциональность, позволяя вводить список [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) файлов. |
| boolean | [**isThreadGLCapable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#isThreadGLCapable())() | Указывает, способен ли текущий поток выполнять работу, связанную с OpenGL. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**removeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#removeGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Удаляет данные [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) из очереди рисования. |
| void | [**setAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setAnimator(com.jogamp.opengl.GLAnimatorControl))​([**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) animatorControl) | Регистрирует использование аниматора, реализующего [GLAnimatorControl](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html). |
| void | [**setAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setAutoSwapBufferMode(boolean))​(boolean enable) | Включает или отключает автоматическую замену буфера для этого объекта рисования. |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**setContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setContext(com.jogamp.opengl.GLContext,boolean))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) newCtx, boolean destroyPrevCtx) | Связывает новый контекст, newtCtx, с этим автоматически отрисовываемым. |
| void | [**setContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setContextCreationFlags(int))​(int flags) |  |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**setExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setExclusiveContextThread(java.lang.Thread))​([**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) t) | Устанавливает этот экземпляр [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) для данного потока. Поток будет запрашивать исключительно [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) посредством вызова [display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())и не освобождать его до тех пор, пока не будет вызван  [destroy()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#destroy())или setExclusiveContextThread(null). |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**setGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setGL(com.jogamp.opengl.GL))​([**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) gl) | Задает объект [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) конвейера, который использует этот GLAutoDrawable. |
| void | [**setGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#setGLEventListenerInitState(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean initialized) | Устанавливает инициализированное состояние [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). |

Методы, унаследованные от GLDrawable

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**createContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#createContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) shareWith) | Создает новый контекст для рисования в этом drawable, который при необходимости будет совместно использовать объекты буфера, текстуры и другие объекты OpenGL на стороне сервера с указанным GLContext. |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getChosenGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getChosenGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html)соответствующие выбранные возможности OpenGL (формат пикселей / визуальный / GLProfile) для этого объекта рисования. |
| [**GLDrawableFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html) | [**getFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getFactory())() | Возвращает значение [GLDrawableFactory](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html), используемой для создания этого экземпляра. |
| [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getGLProfile())() | Возвращает [GLProfile](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html). |
| long | [**getHandle**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getHandle())() | Возвращает дескриптор GL drawable, который гарантированно будет действительным после [realization](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)) *и* во время его [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)[locked](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()) создания. |
| [**NativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html) | [**getNativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface())() | Возвращает связанный [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)с этим [NativeSurfaceHolder](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurfaceHolder.html). |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getRequestedGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getRequestedGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html)соответствующие запрошенным пользователем возможностям OpenGL (формат пикселей / визуальный / GLProfile) для этого объекта рисования. |
| int | [**getSurfaceHeight**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getSurfaceHeight())() | Возвращает высоту [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) клиентской области в пикселях. |
| int | [**getSurfaceWidth**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getSurfaceWidth())() | Возвращает ширину [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) клиентской области в пикселях. |
| boolean | [**isGLOriented**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#isGLOriented())() | Возвращает true, если объект рисования отображается в системе координат OpenGL, начало *координат внизу слева*. |
| boolean | [**isRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#isRealized())() | Возвращает true, если возможность рисования реализована, в противном случае false . |
| void | [**setRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean))​(boolean realized) | Указывает реализациям [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html) GLDrawable, был ли создан базовый компонент и может ли он быть использован. |
| void | [**swapBuffers**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#swapBuffers())() | Меняет местами передний и задний буферы. |

Инициализация GLAutoDrawable

Реализация должна инициализироваться как можно скорее, что возможно только *после* того, как прикрепленное [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)станет видимым и будет реализовано.  
Должна быть реализована следующая последовательность инициализации:

* Создайте [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) с запрошенным [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html)
* Уведомлять [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) для проверки [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html) путем вызова [setRealized(true)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)).
* Создайте новый [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html).
* Инициализируйте все ресурсы OpenGL, вызывая [init(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#init(com.jogamp.opengl.GLAutoDrawable))все зарегистрированные ресурсы [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). Это можно сделать немедленно или с помощью последующего [display(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())вызова.
* Отправьте событие изменения формы, вызвав [reshape(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#reshape(com.jogamp.opengl.GLAutoDrawable,int,int,int,int))всем зарегистрированным [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). Это должно быть сделано после вызова [init(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#init(com.jogamp.opengl.GLAutoDrawable)).

Реконфигурация GLAutoDrawable

Другой деталью реализации является реконфигурация [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html). Одним из вариантов использования является перетаскивание окна на другой экран с другой конфигурацией пикселей, [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html), т. е. Реализация должна быть способна обнаруживать такие случаи в сочетании с соответствующими [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html).  
Например, AWT [Canvas](http://docs.oracle.com/javase/6/docs/api/java/awt/Canvas.html?is-external=true)[getGraphicsConfiguration()](http://docs.oracle.com/javase/6/docs/api/java/awt/Component.html?is-external=true#getGraphicsConfiguration())способен определять изменение устройства отображения. Это продемонстрировано в рамках [GLCanvas](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html) специализации и NEWTAWTCanvas [getGraphicsConfiguration()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGraphicsConfiguration()). Другой демонстрацией является [NativeWindow](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeWindow.html) реализация NEWT на платформе Windows, которая использует функцию MonitorFromWindow *(HWND) собственной платформы.*  
Все ресурсы OpenGL должны быть восстановлены, в то время как drawable [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html) должен быть выбран снова. Должен выполняться следующий протокол.

* Контролируемая утилизация:
  + Утилизируйте все ресурсы OpenGL, вызывая [dispose(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#dispose(com.jogamp.opengl.GLAutoDrawable))всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html).
  + Уничтожьте [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html).
  + Уведомите [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) о недопустимом состоянии путем вызова [setRealized(false)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)).
* Контролируемая регенерация:
  + Создайте новый [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) с запрошенным [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html)
  + Уведомить [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) о повторной проверке [GLCapabilities](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html) путем вызова [setRealized(true)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)).
  + Создайте новый [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html).
  + Инициализируйте все ресурсы OpenGL, вызывая [init(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#init(com.jogamp.opengl.GLAutoDrawable))всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). Это можно сделать немедленно или с помощью последующего вызова [display(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display()).
  + Отправьте событие изменения формы, вызвав [reshape(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#reshape(com.jogamp.opengl.GLAutoDrawable,int,int,int,int))всем зарегистрированным [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html). Это должно быть сделано после вызовов [init(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#init(com.jogamp.opengl.GLAutoDrawable)).

Блокировка GLAutoDrawable

Реализации GLAutoDrawable выполняют блокировку в следующем порядке:

1. [getUpstreamLock()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamLock()).[lock()](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/util/locks/Lock.html?is-external=true#lock())
2. [GLDrawable.getNativeSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()).[lockSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface())

и соответственно снимает блокировки:

1. [GLDrawable.getNativeSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()).[unlockSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#unlockSurface())
2. [getUpstreamLock()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamLock()).[unlock()](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/util/locks/Lock.html?is-external=true#unlock())

Вышеуказанный *порядок блокировки* является обязательным, чтобы гарантировать атомарность работы и избежать условий гонки. Пользовательская реализация или пользовательские приложения, требующие эксклюзивного доступа, должны следовать *порядку блокировки*. См.:

* [getUpstreamLock()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamLock())
* [invoke(boolean, GLRunnable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable))
* [invoke(boolean, List)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,java.util.List))

**Классы GLCanvas и GLJPanel**

GLCanvas и GLJPanel - это два основных класса графического интерфейса JOGL, которые реализуют интерфейс GLAutoDrawable, и которые можно использовать в качестве поверхностей рисования для команд OpenGL.

GLCanvas - это тяжеловесный AWT-компонент, который обеспечивает поддержку рендеринга OpenGL. Это основная реализация интерфейса AWTAutoGLDrawable. Он является потомком класса java.awt.Canvas.

Поскольку это тяжеловесный компонент, в некоторых случаях GLJCanvas может неправильно интегрироваться с компонентами Swing.  Если возникают проблемы с использованием GLCanvas, необходимо использовать класс GLJPanel.

**Класс GLCanvas**

Конструкторы GLCanvas

|  |  |
| --- | --- |
| **Конструктор** | **Описание** |
| [**GLCanvas**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#%3Cinit%3E())() | Создает новый компонент GLCanvas с набором возможностей OpenGL по умолчанию, используя механизм выбора возможностей OpenGL по умолчанию, на экранном устройстве по умолчанию. |
| [**GLCanvas**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#%3Cinit%3E(com.jogamp.opengl.GLCapabilitiesImmutable))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) capsReqUser) | Создает новый компонент GLCanvas с запрошенным набором возможностей OpenGL, используя механизм выбора возможностей OpenGL по умолчанию, на экранном устройстве по умолчанию. |
| [**GLCanvas**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#%3Cinit%3E(com.jogamp.opengl.GLCapabilitiesImmutable,com.jogamp.opengl.GLCapabilitiesChooser,java.awt.GraphicsDevice))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) capsReqUser, [**GLCapabilitiesChooser**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesChooser.html) chooser, [**GraphicsDevice**](http://docs.oracle.com/javase/6/docs/api/java/awt/GraphicsDevice.html?is-external=true) device) | Создает новый компонент GLCanvas. |

Методы GLCanvas

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#addGLEventListener(int,com.jogamp.opengl.GLEventListener))​(int index, [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет данные [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу в эту очередь для рисования. |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#addGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет заданное значение [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) в конец этой очереди рисования. |
| void | [**addNotify**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#addNotify())() | Переопределен для отслеживания добавления этого компонента в контейнер. |
| boolean | [**areAllGLEventListenerInitialized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#areAllGLEventListenerInitialized())() | Возвращает true, если все добавленные [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) инициализированы, в противном случае false . |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**createContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#createContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) shareWith) | Создает новый контекст для рисования для этого объекта рисования, который при необходимости будет совместно использовать объекты буфера, текстуры и другие объекты OpenGL на стороне сервера с указанным GLContext. |
| void | [**destroy**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#destroy())() | Уничтожает все ресурсы, связанные с этим GLAutoDrawable, включая GLContext. |
| void | [**display**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#display())() | Вызывает выполнение рендеринга OpenGL для этого GLAutoDrawable в следующем порядке: вызов [display(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)) всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)файлов. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**disposeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#disposeGLEventListener(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean remove) | Удаляет указанный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) с помощью [dispose(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#dispose(com.jogamp.opengl.GLAutoDrawable)), если он был инициализирован и добавлен в эту очередь. |
| void | [**flushGLRunnables**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#flushGLRunnables())() | Сбрасывает все [enqueued](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable)) [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) этого [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), включая уведомление ожидающего исполнителя. |
| [**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) | [**getAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getAnimator())() |  |
| boolean | [**getAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getAutoSwapBufferMode())() | Указывает, включена ли автоматическая замена буфера для этого объекта рисования. |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getChosenGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getChosenGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html)соответствующие возможности OpenGL |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**getContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getContext())() | Возвращает контекст, связанный с этим объектом рисования. |
| int | [**getContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getContextCreationFlags())() |  |
| float[] | [**getCurrentSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getCurrentSurfaceScale(float%5B%5D))​(float[] result) | Возвращает текущий масштаб пикселя связанного с [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| [**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) | [**getDefaultCloseOperation**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getDefaultCloseOperation())() |  |
| [**GLDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) | [**getDelegatedDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getDelegatedDrawable())() | Если реализация использует делегирование, возвращает делегированный экземпляр [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html), в противном случае возвращает this. |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**getExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getExclusiveContextThread())() |  |
| [**GLDrawableFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html) | [**getFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getFactory())() | Возвращает значение [GLDrawableFactory](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html), использованное для создания этого экземпляра. |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**getGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGL())() | Возвращает объект [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) конвейера, который использует этот GLAutoDrawable. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**getGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGLEventListener(int))​(int index) | Возвращает значение [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу этой очереди для рисования. |
| int | [**getGLEventListenerCount**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGLEventListenerCount())() | Возвращает номер [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) этой очереди для рисования. |
| boolean | [**getGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGLEventListenerInitState(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Извлекает, инициализирован ли данный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) или нет. |
| [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGLProfile())() | Возвращает [GLProfile](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) |
| [**GraphicsConfiguration**](http://docs.oracle.com/javase/6/docs/api/java/awt/GraphicsConfiguration.html?is-external=true) | [**getGraphicsConfiguration**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getGraphicsConfiguration())() |  |
| long | [**getHandle**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getHandle())() | Возвращает дескриптор GL, который гарантированно будет действительным после [realization](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)) *и* во время его [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)[locked](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()) создания. |
| float[] | [**getMaximumSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getMaximumSurfaceScale(float%5B%5D))​(float[] result) | Возвращает максимальный масштаб пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| float[] | [**getMinimumSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getMinimumSurfaceScale(float%5B%5D))​(float[] result) | Возвращает минимальный масштаб пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| [**NativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html) | [**getNativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getNativeSurface())() | Возвращает связанный [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)с этим [NativeSurfaceHolder](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurfaceHolder.html). |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getRequestedGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getRequestedGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html)соответствующие запрошенным пользователем возможностям OpenGL |
| float[] | [**getRequestedSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getRequestedSurfaceScale(float%5B%5D))​(float[] result) | Возвращает масштаб [requested](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/ScalableSurface.html#setSurfaceScale(float%5B%5D)) пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| boolean | [**getShallUseOffscreenLayer**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getShallUseOffscreenLayer())() | Возвращает свойство, установленное с помощью [OffscreenLayerOption.setShallUseOffscreenLayer(boolean)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/OffscreenLayerOption.html#setShallUseOffscreenLayer(boolean)). |
| int | [**getSurfaceHeight**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getSurfaceHeight())() | Возвращает высоту [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) клиентской области в пикселях. |
| int | [**getSurfaceWidth**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getSurfaceWidth())() | Возвращает ширину [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) клиентской области в пикселях. |
| [**RecursiveLock**](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/util/locks/RecursiveLock.html?is-external=true) | [**getUpstreamLock**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getUpstreamLock())() | Возвращает рекурсивный объект блокировки [upstream widget](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamWidget()) для синхронизации многопоточного доступа поверх [NativeSurface.lockSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()). |
| [**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) | [**getUpstreamWidget**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#getUpstreamWidget())() | Метод *может* возвращать вышестоящий объект инструментария пользовательского интерфейса, содержащий этот экземпляр [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), если он существует. |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#invoke(boolean,com.jogamp.opengl.GLRunnable))​(boolean wait, [**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) glRunnable) | Ставит в очередь одноразовый [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) запрос, который будет выполнен в рамках следующего [GLAutoDrawable.display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())вызова после вызова всех зарегистрированных методов [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)[display(GLAutoDrawable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)). |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#invoke(boolean,java.util.List))​(boolean wait, [**List**](http://docs.oracle.com/javase/6/docs/api/java/util/List.html?is-external=true)<[**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html)> glRunnables) | Расширяет [GLAutoDrawable.invoke(boolean, GLRunnable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable))функциональность, позволяя вводить список [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) файлов. |
| boolean | [**isGLOriented**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#isGLOriented())() | Возвращает true, если отрисовываемый объект отображается в системе координат OpenGL, начало *координат внизу слева*. |
| boolean | [**isOffscreenLayerSurfaceEnabled**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#isOffscreenLayerSurfaceEnabled())() | Возвращает true, если этот экземпляр использует закадровый слой, в противном случае false. |
| boolean | [**isRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#isRealized())() | Возвращаетtrue, если эта возможность рисования реализована, в противном случае false . |
| boolean | [**isThreadGLCapable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#isThreadGLCapable())() | Указывает, способен ли текущий поток выполнять работу, связанную с OpenGL. |
| static void | [**main**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#main(java.lang.String%5B%5D))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] args) | Самая простая тестовая запись JOGL AWT |
| void | [**paint**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#paint(java.awt.Graphics))​([**Graphics**](http://docs.oracle.com/javase/6/docs/api/java/awt/Graphics.html?is-external=true) g) | Переопределен, чтобы вызвать выполнение рендеринга OpenGL во время циклов перерисовки. |
| void | [**print**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#print(java.awt.Graphics))​([**Graphics**](http://docs.oracle.com/javase/6/docs/api/java/awt/Graphics.html?is-external=true) graphics) |  |
| void | [**releasePrint**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#releasePrint())() | Должны быть вызваны после [PrinterJob.print()](http://docs.oracle.com/javase/6/docs/api/java/awt/print/PrinterJob.html?is-external=true#print()). |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**removeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#removeGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Удаляет данные [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)из этой очереди рисования. |
| void | [**removeNotify**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#removeNotify())() | Переопределяется для отслеживания, когда этот компонент удаляется из контейнера. |
| void | [**reshape**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#reshape(int,int,int,int))​(int x, int y, int width, int height) | Переопределен, чтобы вызвать GLDrawableHelper.reshape(com.jogamp.opengl.GLAutoDrawable, int, int, int, int) вызов для всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)файлов. |
| void | [**setAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setAnimator(com.jogamp.opengl.GLAnimatorControl))​([**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) animatorControl) | Регистрирует использование аниматора, реализацию [GLAnimatorControl](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html). |
| void | [**setAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setAutoSwapBufferMode(boolean))​(boolean onOrOff) | Включает или отключает автоматическую замену буфера для этого объекта рисования. |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**setContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setContext(com.jogamp.opengl.GLContext,boolean))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) newCtx, boolean destroyPrevCtx) | Связывает новый контекст, newtCtx, с этим автоматически извлекаемым. |
| void | [**setContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setContextCreationFlags(int))​(int flags) |  |
| [**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) | [**setDefaultCloseOperation**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setDefaultCloseOperation(com.jogamp.nativewindow.WindowClosingProtocol.WindowClosingMode))​([**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) op) |  |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**setExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setExclusiveContextThread(java.lang.Thread))​([**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) t) | Выделяет этот экземпляр [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) для данного потока. Поток будет запрашивать исключительно [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html)via [GLAutoDrawable.display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())и не освобождать его до [GLAutoDrawable.destroy()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#destroy())тех пор, пока не будет вызван или setExclusiveContextThread(null)не будет вызван. |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**setGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setGL(com.jogamp.opengl.GL))​([**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) gl) | Задает объект [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) конвейера, который использует этот GLAutoDrawable. |
| void | [**setGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setGLEventListenerInitState(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean initialized) | Устанавливает заданное инициализированное состояние [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) . |
| void | [**setRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setRealized(boolean))​(boolean realized) | Указывает для реализаций [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html) GLDrawable, был ли создан базовый компонент и может ли он быть использован. |
| void | [**setShallUseOffscreenLayer**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setShallUseOffscreenLayer(boolean))​(boolean v) | Возвращает закадровый слой, если он поддерживается. |
| void | [**setSharedAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setSharedAutoDrawable(com.jogamp.opengl.GLAutoDrawable))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) sharedAutoDrawable) | Указывает [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), который [OpenGL context](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) должен быть общим [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) для этого [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html). |
| void | [**setSharedContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setSharedContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) sharedContext) | Указывает [OpenGL context](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html), который должен быть общим [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) для этого [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html). |
| boolean | [**setSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setSurfaceScale(float%5B%5D))​(float[] pixelScale) | Запрашивать масштаб пикселей в направлениях x и y для связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html), где size\_in\_pixel\_units = pixel\_scale \* size\_in\_window\_units. |
| void | [**setupPrint**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#setupPrint(double,double,int,int,int))​(double scaleMatX, double scaleMatY, int numSamples, int tileWidth, int tileHeight) | Должен быть вызван раньше [PrinterJob.print()](http://docs.oracle.com/javase/6/docs/api/java/awt/print/PrinterJob.html?is-external=true#print()). |
| void | [**swapBuffers**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#swapBuffers())() | Меняет местами передний и задний буферы. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**toString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#toString())() |  |
| void | [**update**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLCanvas.html#update(java.awt.Graphics))​([**Graphics**](http://docs.oracle.com/javase/6/docs/api/java/awt/Graphics.html?is-external=true) g) | Переопределен из Canvas, чтобы предотвратить очистку canvas AWT от вмешательства в рендеринг OpenGL. |

Класс GLJPanel

Конструкторы GLJPanel

|  |  |
| --- | --- |
| **Конструктор** | **Описание** |
| [**GLJPanel**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#%3Cinit%3E())() | Создает новый компонент GLJPanel с набором возможностей OpenGL по умолчанию и с использованием механизма выбора возможностей OpenGL по умолчанию. |
| [**GLJPanel**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#%3Cinit%3E(com.jogamp.opengl.GLCapabilitiesImmutable))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) userCapsRequest) | Создает новый компонент GLJPanel с запрошенным набором возможностей OpenGL, используя механизм выбора возможностей OpenGL по умолчанию. |
| [**GLJPanel**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#%3Cinit%3E(com.jogamp.opengl.GLCapabilitiesImmutable,com.jogamp.opengl.GLCapabilitiesChooser))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) userCapsRequest, [**GLCapabilitiesChooser**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesChooser.html) chooser) | Создает новый компонент GLJPanel. |

Методы GLJPanel

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#addGLEventListener(int,com.jogamp.opengl.GLEventListener))​(int index, [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет данные [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу в эту очередь для рисования. |
| void | [**addGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#addGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Добавляет заданное значение [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) в конец этой очереди рисования. |
| void | [**addNotify**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#addNotify())() | Переопределяется для отслеживания добавления этого компонента в контейнер. |
| boolean | [**areAllGLEventListenerInitialized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#areAllGLEventListenerInitialized())() | Возвращает true, если все добавленные [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) инициализированы, в противном случае false . |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**createContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#createContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) shareWith) | Создает новый контекст для рисования для этого объекта рисования, который при необходимости будут совместно использовать объекты буфера, текстуры и другие объекты OpenGL на стороне сервера с указанным GLContext. |
| void | [**destroy**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#destroy())() | Просто псевдоним для removeNotify |
| void | [**display**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#display())() | Вызывает выполнение рендеринга OpenGL для этого GLAutoDrawable в следующем порядке: вызов [display(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)) всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) файлов. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**disposeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#disposeGLEventListener(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean remove) | Удаляет указанный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) через [dispose(..)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#dispose(com.jogamp.opengl.GLAutoDrawable)), если он был инициализирован и добавлен в эту очередь. |
| void | [**flushGLRunnables**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#flushGLRunnables())() | Сбрасывает все [enqueued](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable)) [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html)это[GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), включая уведомление ожидающего исполнителя. |
| [**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) | [**getAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getAnimator())() |  |
| boolean | [**getAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getAutoSwapBufferMode())() | Указывает, включена ли автоматическая замена буфера для этого объекта рисования. |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getChosenGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getChosenGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) соответствующие выбранным возможностям OpenGL |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**getContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getContext())() | Возвращает контекст, связанный с этим объектом рисования. |
| int | [**getContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getContextCreationFlags())() |  |
| float[] | [**getCurrentSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getCurrentSurfaceScale(float%5B%5D))​(float[] result) | Возвращает текущий масштаб пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| [**AWTGLPixelBuffer.AWTGLPixelBufferProvider**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/util/awt/AWTGLPixelBuffer.AWTGLPixelBufferProvider.html) | [**getCustomPixelBufferProvider**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getCustomPixelBufferProvider())() |  |
| [**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) | [**getDefaultCloseOperation**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getDefaultCloseOperation())() |  |
| [**GLDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html) | [**getDelegatedDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getDelegatedDrawable())() | Если реализация использует делегирование, возвращает делегированный экземпляр [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html), в противном случае this. |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**getExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getExclusiveContextThread())() |  |
| [**GLDrawableFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html) | [**getFactory**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getFactory())() | Возвращает значение [GLDrawableFactory](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html), используемое для создания этого экземпляра. |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**getGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getGL())() | Возвращает объект [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) конвейера, используемый этим GLAutoDrawable. |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**getGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getGLEventListener(int))​(int index) | Возвращает значение [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) по заданному индексу этой очереди рисования. |
| int | [**getGLEventListenerCount**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getGLEventListenerCount())() | Возвращает номер [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) этой очереди для рисования. |
| boolean | [**getGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getGLEventListenerInitState(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Извлекает, инициализирован ли данный [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) или нет. |
| [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getGLProfile())() | Извлекает [GLProfile](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) |
| long | [**getHandle**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getHandle())() | Возвращает дескриптор GL, который гарантированно будет действительным после [realization](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#setRealized(boolean)) *и* во время его создания [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)[locked](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()). |
| float[] | [**getMaximumSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getMaximumSurfaceScale(float%5B%5D))​(float[] result) | Возвращает максимальный масштаб пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| float[] | [**getMinimumSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getMinimumSurfaceScale(float%5B%5D))​(float[] result) | Возвращает минимальный масштаб пикселя связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| [**NativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html) | [**getNativeSurface**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getNativeSurface())() | Возвращает значение , связанное [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)с этим [NativeSurfaceHolder](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurfaceHolder.html). |
| [**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) | [**getRequestedGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getRequestedGLCapabilities())() | Извлекает [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) соответствующие запрошенным пользователем возможностям OpenGL |
| float[] | [**getRequestedSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getRequestedSurfaceScale(float%5B%5D))​(float[] result) | Возвращает [requested](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/ScalableSurface.html#setSurfaceScale(float%5B%5D)) пиксельный масштаб связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html). |
| boolean | [**getSkipGLOrientationVerticalFlip**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getSkipGLOrientationVerticalFlip())() | Смотри [setSkipGLOrientationVerticalFlip(boolean)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setSkipGLOrientationVerticalFlip(boolean)). |
| int | [**getSurfaceHeight**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getSurfaceHeight())() | Возвращает высоту клиентской области [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) в пикселях. |
| int | [**getSurfaceWidth**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getSurfaceWidth())() | Возвращает ширину клиентской области [GLDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html)[surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawable.html#getNativeSurface()) в пикселях. |
| int | [**getTextureUnit**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getTextureUnit())() | Возвращает используемую текстурную единицу. |
| [**RecursiveLock**](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/util/locks/RecursiveLock.html?is-external=true) | [**getUpstreamLock**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getUpstreamLock())() | Возвращает рекурсивный объект блокировки [upstream widget](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#getUpstreamWidget()) для синхронизации многопоточного доступа поверх [NativeSurface.lockSurface()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html#lockSurface()). |
| [**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) | [**getUpstreamWidget**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#getUpstreamWidget())() | Метод *может* возвращать вышестоящий объект инструментария пользовательского интерфейса, содержащий этот экземпляр [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), если он существует. |
| boolean | [**initializeBackend**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#initializeBackend(boolean))​(boolean offthread) | Пытается инициализировать серверную часть, если она еще не инициализирована. |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#invoke(boolean,com.jogamp.opengl.GLRunnable))​(boolean wait, [**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html) glRunnable) | Ставит в очередь одноразовый запрос [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html), который будет выполнен в рамках следующего [GLAutoDrawable.display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())вызова после вызова всех зарегистрированных методов [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html)[display(GLAutoDrawable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html#display(com.jogamp.opengl.GLAutoDrawable)). |
| boolean | [**invoke**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#invoke(boolean,java.util.List))​(boolean wait, [**List**](http://docs.oracle.com/javase/6/docs/api/java/util/List.html?is-external=true)<[**GLRunnable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html)> glRunnables) | Расширяет функциональность [GLAutoDrawable.invoke(boolean, GLRunnable)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#invoke(boolean,com.jogamp.opengl.GLRunnable)), позволяя вводить список файлов [GLRunnable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLRunnable.html). |
| boolean | [**isGLOriented**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#isGLOriented())() | Возвращает true, если отрисовываемый объект отображается в системе координат OpenGL, начало *координат внизу слева*. |
| boolean | [**isRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#isRealized())() | Возвращает true, если эта возможность рисования реализована, в противном случае false . |
| boolean | [**isThreadGLCapable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#isThreadGLCapable())() | Указывает, способен ли текущий поток выполнять работу, связанную с OpenGL. |
| void | [**print**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#print(java.awt.Graphics))​([**Graphics**](http://docs.oracle.com/javase/6/docs/api/java/awt/Graphics.html?is-external=true) graphics) |  |
| void | [**releasePrint**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#releasePrint())() | Должен быть вызван после [PrinterJob.print()](http://docs.oracle.com/javase/6/docs/api/java/awt/print/PrinterJob.html?is-external=true#print()). |
| [**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) | [**removeGLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#removeGLEventListener(com.jogamp.opengl.GLEventListener))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener) | Удаляет данные [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) из этой очереди рисования. |
| void | [**removeNotify**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#removeNotify())() | Переопределяется для отслеживания, когда этот компонент удаляется из контейнера. |
| void | [**reshape**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#reshape(int,int,int,int))​(int x, int y, int width, int height) | Переопределено, чтобы вызвать GLDrawableHelper.reshape(com.jogamp.opengl.GLAutoDrawable, int, int, int, int)вызов для всех зарегистрированных [GLEventListener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) файлов. |
| void | [**setAnimator**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setAnimator(com.jogamp.opengl.GLAnimatorControl))​([**GLAnimatorControl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html) animatorControl) | Регистрирует использование аниматора, реализацию [GLAnimatorControl](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAnimatorControl.html). |
| void | [**setAutoSwapBufferMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setAutoSwapBufferMode(boolean))​(boolean enable) | Включает или отключает автоматическую замену буфера для этого чертежа. |
| [**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) | [**setContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setContext(com.jogamp.opengl.GLContext,boolean))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) newCtx, boolean destroyPrevCtx) | Связывает новый контекст, newtCtx, с этим автоматически извлекаемым. |
| void | [**setContextCreationFlags**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setContextCreationFlags(int))​(int flags) |  |
| [**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) | [**setDefaultCloseOperation**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setDefaultCloseOperation(com.jogamp.nativewindow.WindowClosingProtocol.WindowClosingMode))​([**WindowClosingProtocol.WindowClosingMode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/WindowClosingProtocol.WindowClosingMode.html) op) |  |
| [**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) | [**setExclusiveContextThread**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setExclusiveContextThread(java.lang.Thread))​([**Thread**](http://docs.oracle.com/javase/6/docs/api/java/lang/Thread.html?is-external=true) t) | Выделяет этот экземпляр [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) для данного потока. Поток будет запрашивать исключительно [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) через [GLAutoDrawable.display()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#display())и не освобождать его до тех пор, пока [GLAutoDrawable.destroy()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html#destroy())не будет вызван или не будет вызван setExclusiveContextThread(null). |
| [**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) | [**setGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setGL(com.jogamp.opengl.GL))​([**GL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) gl) | Задает объект [GL](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GL.html) конвейера, который использует этот GLAutoDrawable. |
| void | [**setGLEventListenerInitState**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setGLEventListenerInitState(com.jogamp.opengl.GLEventListener,boolean))​([**GLEventListener**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) listener, boolean initialized) | Устанавливает заданное [listener](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLEventListener.html) инициализированное состояние. |
| void | [**setOpaque**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setOpaque(boolean))​(boolean opaque) |  |
| void | [**setPixelBufferProvider**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setPixelBufferProvider(com.jogamp.opengl.util.awt.AWTGLPixelBuffer.AWTGLPixelBufferProvider))​([**AWTGLPixelBuffer.AWTGLPixelBufferProvider**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/util/awt/AWTGLPixelBuffer.AWTGLPixelBufferProvider.html) custom) |  |
| void | [**setRealized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setRealized(boolean))​(boolean realized) | Указывает для реализаций [surface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html)GLDrawable, был ли создан базовый компонент и может ли он быть использован. |
| void | [**setRequestedGLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setRequestedGLCapabilities(com.jogamp.opengl.GLCapabilitiesImmutable))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) caps) | Задайте новый запрос [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) для этой GLJPanel, разрешающий реконфигурацию. |
| void | [**setSharedAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setSharedAutoDrawable(com.jogamp.opengl.GLAutoDrawable))​([**GLAutoDrawable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) sharedAutoDrawable) | Указывает [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html), который [OpenGL context](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html)должен быть общим [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) для этого [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html). |
| void | [**setSharedContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setSharedContext(com.jogamp.opengl.GLContext))​([**GLContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html) sharedContext) | Указывает an [OpenGL context](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html), который должен быть общим [GLAutoDrawable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLAutoDrawable.html) для этого [GLContext](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLContext.html). |
| void | [**setSkipGLOrientationVerticalFlip**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setSkipGLOrientationVerticalFlip(boolean))​(boolean v) | [isGLOriented()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#isGLOriented())Вертикальный флип на основе пропуска, который обычно требуется серверной частью вне экрана,. |
| boolean | [**setSurfaceScale**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setSurfaceScale(float%5B%5D))​(float[] pixelScale) | Запрашивайте масштаб пикселей в направлениях x и y для связанного [NativeSurface](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/NativeSurface.html), где size\_in\_pixel\_units = pixel\_scale \* size\_in\_window\_units. |
| void | [**setTextureUnit**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setTextureUnit(int))​(int v) | Позволяет пользователю запрашивать текстурный модуль для использования, который должен быть вызван перед первой инициализацией |
| void | [**setupPrint**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setupPrint(double,double,int,int,int))​(double scaleMatX, double scaleMatY, int numSamples, int tileWidth, int tileHeight) | Должен быть вызван раньше [PrinterJob.print()](http://docs.oracle.com/javase/6/docs/api/java/awt/print/PrinterJob.html?is-external=true#print()). |
| boolean | [**shouldPreserveColorBufferIfTranslucent**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#shouldPreserveColorBufferIfTranslucent())() | Для полупрозрачной GLJPanel (для которой [setOpaque](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#setOpaque(boolean)) было вызвано (false)) указывает, должно ли приложение сохранять буфер цвета OpenGL (GL\_COLOR\_BUFFER\_BIT) для корректного отображения GLJPanel и базовых виджетов, которые могут отображаться через части GLJPanel со значениями альфа меньше 1. |
| void | [**swapBuffers**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#swapBuffers())() | Меняет местами передний и задний буферы |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**toString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/awt/GLJPanel.html#toString())() |  |

**Класс GLCapabilities**

Чтобы создать экземпляр классов GLCanvas или GLJPanel, необходим объект класса, реализующего интерфейс GLCapabilitiesImmutable, который определяет неизменяемый (immutable) набор возможностей OpenGL.

Один из способов получить объект интерфейса CapabilitiesImmutable - создать экземпляр класса GLCapabilities, который реализует этот интерфейс.

Класс GLCapabilities описывает желаемые возможности, которые должен поддерживать контекст рендеринга, например, профиль OpenGL.

Конструктор класса GLCapabilities

|  |  |
| --- | --- |
| **Конструктор** | **Описание** |
| [**GLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#%3Cinit%3E(com.jogamp.opengl.GLProfile))​([**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) glp) | Создает объект GLCapabilities. |

Методы класса GLCapabilities

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| [**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) | [**clone**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#clone())() |  |
| [**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) | [**cloneMutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#cloneMutable())() |  |
| int | [**compareTo**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#compareTo(com.jogamp.nativewindow.CapabilitiesImmutable))​([**CapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/CapabilitiesImmutable.html) o) | только сравнение hw / sw, стерео, мультисэмплов, трафаретов, RGBA и глубины |
| [**GLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html) | [**copyFrom**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#copyFrom(com.jogamp.nativewindow.CapabilitiesImmutable))​([**CapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/CapabilitiesImmutable.html) source) | Копирует все [CapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/CapabilitiesImmutable.html) значения из source этого экземпляра. |
| [**GLCapabilities**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html) | [**copyFrom**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#copyFrom(com.jogamp.opengl.GLCapabilitiesImmutable))​([**GLCapabilitiesImmutable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) source) | Копирует все [GLCapabilitiesImmutable](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html) значения из source этого экземпляра. |
| boolean | [**equals**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#equals(java.lang.Object))​([**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) obj) | Равенство по неизменяемым атрибутам обоих объектов |
| int | [**getAccumAlphaBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getAccumAlphaBits())() | Возвращает количество бит для альфа-компонента буфера накопления. |
| int | [**getAccumBlueBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getAccumBlueBits())() | Возвращает количество бит для синего компонента накопительного буфера. |
| int | [**getAccumGreenBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getAccumGreenBits())() | Возвращает количество бит для зеленого компонента накопительного буфера. |
| int | [**getAccumRedBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getAccumRedBits())() | Возвращает количество битов для красного компонента буфера накопления. |
| int | [**getDepthBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getDepthBits())() | Возвращает количество битов буфера глубины. |
| boolean | [**getDoubleBuffered**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getDoubleBuffered())() | Возвращает, запрошена ли двойная буферизация, доступна или выбрана. |
| [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getGLProfile())() | Возвращает профиль GL, который вы хотите или который используется для рисования. |
| boolean | [**getHardwareAccelerated**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getHardwareAccelerated())() | Возвращает, запрошено ли аппаратное ускорение, доступно или выбрано. |
| int | [**getNumSamples**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getNumSamples())() | Возвращает количество буферов выборки, которые должны быть выделены, если буферы выборки включены, в противном случае возвращает 0. |
| boolean | [**getSampleBuffers**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getSampleBuffers())() | Возвращает, следует ли выделять буферы выборки для сглаживания всей сцены (FSAA) для этого чертежа. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getSampleExtension**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getSampleExtension())() | Возвращает расширение для сглаживания всей сцены (FSAA). |
| int | [**getStencilBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getStencilBits())() | Возвращает количество битов буфера трафарета. |
| boolean | [**getStereo**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#getStereo())() | Возвращает, запрошен ли stereo, доступен или выбран. |
| int | [**hashCode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#hashCode())() | хэш-код над неизменяемыми атрибутами обоих объектов |
| boolean | [**isFBO**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#isFBO())() | Возвращает, запрошен ли, доступен или выбран ли закадровый режим FBO. |
| boolean | [**isPBuffer**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#isPBuffer())() | Возвращает, запрошен ли, доступен или выбран закадровый режим pbuffer. |
| void | [**setAccumAlphaBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setAccumAlphaBits(int))​(int accumAlphaBits) | Задает количество битов, запрашиваемых для альфа-компонента накопительного буфера. |
| void | [**setAccumBlueBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setAccumBlueBits(int))​(int accumBlueBits) | Задает количество битов, запрашиваемых для синего компонента накопительного буфера. |
| void | [**setAccumGreenBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setAccumGreenBits(int))​(int accumGreenBits) | Задает количество битов, запрашиваемых для зеленого компонента накопительного буфера. |
| void | [**setAccumRedBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setAccumRedBits(int))​(int accumRedBits) | Задает количество битов, запрашиваемых для красного компонента накопительного буфера. |
| void | [**setDepthBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setDepthBits(int))​(int depthBits) | Задает количество битов, запрашиваемых для буфера глубины. |
| void | [**setDoubleBuffered**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setDoubleBuffered(boolean))​(boolean enable) | Включает или отключает двойную буферизацию. |
| void | [**setFBO**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setFBO(boolean))​(boolean enable) | Запрашивает закадровый режим FBO. |
| void | [**setGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setGLProfile(com.jogamp.opengl.GLProfile))​([**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) profile) | Устанавливает желаемый профиль GL |
| void | [**setHardwareAccelerated**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setHardwareAccelerated(boolean))​(boolean enable) | Включает или отключает аппаратное ускорение. |
| void | [**setNumSamples**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setNumSamples(int))​(int numSamples) | Если буферы выборки включены, указывает количество буферов, которые должны быть выделены. |
| void | [**setPBuffer**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setPBuffer(boolean))​(boolean enable) | Запрашивает закадровый режим pbuffer. |
| void | [**setSampleBuffers**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setSampleBuffers(boolean))​(boolean enable) | По умолчанию используется значение false. Указывает, следует ли выделять буферы выборки для сглаживания всей сцены (FSAA) для этого чертежа. Имейте в виду, что для этого требуется альфа-компонент. Если этот метод включен, он также вызывает [setAlphaBits(1)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/Capabilities.html#setAlphaBits(int))если [Capabilities.getAlphaBits()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/Capabilities.html#getAlphaBits())== 0. |
| void | [**setSampleExtension**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setSampleExtension(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) se) | Задает желаемое расширение для сглаживания всей сцены (FSAA), по умолчанию [GLCapabilitiesImmutable.DEFAULT\_SAMPLE\_EXTENSION](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilitiesImmutable.html#DEFAULT_SAMPLE_EXTENSION). |
| void | [**setStencilBits**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setStencilBits(int))​(int stencilBits) | Задает количество битов, запрашиваемых для буфера трафарета. |
| void | [**setStereo**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#setStereo(boolean))​(boolean enable) | Включает или отключает просмотр стерео. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**toString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#toString())() | Возвращает текстовое представление этого объекта GLCapabilities. |
| [**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) | [**toString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLCapabilities.html#toString(java.lang.StringBuilder))​([**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) sink) | Возвращает текстовое представление этого объекта. |

**Класс GLProfile**

Поскольку было выпущено несколько версий OpenGL API, то необходимо указать версию OpenGL API, используемую в вашей программе. Это делается с помощью класса GLProfile.

Класс GLProfile задает профиль OpenGL. Статическая одноэлементная инициализация этого класса запрашивает доступность всех профилей OpenGL и создает экземпляры одноэлементных объектов GLProfile для каждого доступного профиля. Профиль платформы по умолчанию может указываться, используя вызов GLProfile.GetProfileDefault(), или более специализированные версии, использующие другие статические методы getProfile.

Поля класса GLProfile

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Поле** | **Описание** |
| static boolean | [**DEBUG**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#DEBUG) |  |
| static boolean | [**disabledEGL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disabledEGL) | В случае, если реализация EGL не доступна, как на платформе [Platform.OSType.IOS](https://jogamp.org/deployment/jogamp-next/javadoc/gluegen/javadoc/com/jogamp/common/os/Platform.OSType.html?is-external=true#IOS) устанавливается значение true. |
| static boolean | [**disableOpenGLARBContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disableOpenGLARBContext) | В случае, если реализация *расширения* для создания контекста ARB\_create\_context дает сбои на платформе, установленое свойство jogl.disable.openglarbcontext отключает его использование. |
| static boolean | [**disableOpenGLCore**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disableOpenGLCore) | В случае, если собственные профили ядра OpenGL не требуются и если одна платформа может иметь ошибочную реализацию, установка свойства jogl.disable.openglcore отключает запрос возможных существующих собственных профилей ядра OpenGL. |
| static boolean | [**disableOpenGLDesktop**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disableOpenGLDesktop) | В случае, если профили рабочего стола OpenGL не требуются и если платформа может иметь ошибочную реализацию, установка свойства jogl.disable.opengldesktop отключает запрос возможных существующих профилей рабочего стола OpenGL. |
| static boolean | [**disableOpenGLES**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disableOpenGLES) | В случае, если профили OpenGL ES не требуются и если платформа может иметь ошибочную реализацию, установка свойства jogl.disable.opengles отключает запрос возможных существующих профилей OpenGL ES. |
| static boolean | [**disableSurfacelessContext**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#disableSurfacelessContext) | Отключите возможность бесповерхностного контекста OpenGL и его зондирования , установив свойство jogl.disable.surfacelesscontext. |
| static boolean | [**enableANGLE**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#enableANGLE) | Необходимо отключить поддержку ANGLE, эмуляции D3D ES2 в Windows, предоставляемой с Firefox и Chrome. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_ALL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_ALL) | Все профили GL в порядке определения по умолчанию. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MAX**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MAX) | Порядок максимального количества профилей. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MAX\_FIXEDFUNC**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MAX_FIXEDFUNC) | Порядок максимальных фиксированных функциональных профилей GL4bc GL3bc GL2 GLES1 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MAX\_MOBILE**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MAX_MOBILE) | Заказ максимального количества оригинальных мобильных профилей. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MAX\_PROGSHADER**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MAX_PROGSHADER) | Порядок максимального количества программируемых шейдерных профилей GL4bc GL4 GL3bc GL3 GLES3 GL2 GLES2 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MAX\_PROGSHADER\_CORE**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MAX_PROGSHADER_CORE) | Порядок максимального количества программируемых шейдерных *ядер профилирует только* GL4 GL3 GLES3 GLES2 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MIN**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MIN) | Порядок минимальных профилей. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] | [**GL\_PROFILE\_LIST\_MIN\_DESKTOP**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL_PROFILE_LIST_MIN_DESKTOP) | Заказ минимального количества оригинальных профилей рабочего стола. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL2) | Рабочий стол OpenGL profile 1.x до версии 3.0 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL2ES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL2ES1) | Пересечение рабочего GL2 и встроенного ES1 профиля |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL2ES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL2ES2) | Пересечение рабочего GL3, GL2 и встроенного профиля ES2 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL2GL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL2GL3) | Пересечение профилей GL3 и GL2 для рабочего стола |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL3) | Рабочий стол OpenGL core profile 3.x, с x>= 1 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL3bc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL3bc) | Профиль совместимости рабочего стола с OpenGL 3.x, с x> = 1, т.Е. GL2 плюс GL3. bc расшифровывается как обратная совместимость. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL4**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL4) | Рабочий стол OpenGL core profile 4.x, с x>= 0 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL4bc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL4bc) | Профиль совместимости рабочего стола с OpenGL 4.x, с x> = 0, т.Е. GL2 плюс GL4. bc расшифровывается как обратная совместимость. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GL4ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GL4ES3) | Пересечение профиля рабочего стола GL4 и ES3, доступное только при наличии ES3 или GL4 w/GL\_ARB\_ES3\_compatibility. |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GLES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GLES1) | Встроенный профиль OpenGL составляет 1.x, при этом x >= 0 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GLES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GLES2) | Встроенный профиль OpenGL ES 2.x, с x>= 0 |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**GLES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#GLES3) | Встроенный профиль OpenGL - это 3.x, с x>= 0 |

Методы класса GLProfile

|  |  |  |
| --- | --- | --- |
| **Модификатор и тип** | **Метод** | **Описание** |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**createCustomGLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#createCustomGLProfile(java.lang.String,com.jogamp.opengl.GLProfile))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profile, [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) profileImpl) |  |
| boolean | [**equals**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#equals(java.lang.Object))​([**Object**](http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html?is-external=true) o) |  |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**get**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#get(com.jogamp.nativewindow.AbstractGraphicsDevice,java.lang.String))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profile) | Возвращает объект GLProfile. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**get**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#get(com.jogamp.nativewindow.AbstractGraphicsDevice,java.lang.String%5B%5D,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] profiles, boolean favorHardwareRasterizer) | Возвращает первый профиль из заданного списка, в котором доступна реализация. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**get**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#get(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profile) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**get**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#get(java.lang.String%5B%5D,boolean))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] profiles, boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getDefault**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getDefault())() | Возвращает объект GLProfile по умолчанию, отражающий наилучший вариант для текущей платформы. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getDefault**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getDefault(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Возвращает объект GLProfile по умолчанию, отражающий наилучший вариант для текущей платформы. |
| static [**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) | [**getDefaultDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getDefaultDevice())() |  |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2ES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES1())() | Вызовы [getGL2ES1(AbstractGraphicsDevice)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES1(com.jogamp.nativewindow.AbstractGraphicsDevice))с использованием устройства по умолчанию. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2ES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES1(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Возвращает реализацию профиля GL2ES1, следовательно, совместимую с GL2ES1. Он возвращается: |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2ES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES2())() | Вызовы [getGL2ES2(AbstractGraphicsDevice)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES2(com.jogamp.nativewindow.AbstractGraphicsDevice))с использованием устройства по умолчанию. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2ES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2ES2(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Возвращает реализацию профиля GL2ES2, следовательно, совместимую с GL2ES2. Он возвращает: |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2GL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2GL3())() | Вызовы [getGL2GL3(AbstractGraphicsDevice)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2GL3(com.jogamp.nativewindow.AbstractGraphicsDevice))с использованием устройства по умолчанию. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL2GL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL2GL3(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Возвращает реализацию профиля GL2GL3, следовательно, совместимую с GL2GL3. Он возвращается: |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL4ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL4ES3())() | Вызовы [getGL4ES3(AbstractGraphicsDevice)](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL4ES3(com.jogamp.nativewindow.AbstractGraphicsDevice))с использованием устройства по умолчанию. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getGL4ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGL4ES3(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Возвращает реализацию профиля GL4ES3, следовательно, совместимую с GL4ES3. Он возвращает: |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getGLArrayName**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGLArrayName(int))​(int array) |  |
| [**Constructor**](http://docs.oracle.com/javase/6/docs/api/java/lang/reflect/Constructor.html?is-external=true)<?> | [**getGLCtor**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGLCtor(boolean))​(boolean glObject) |  |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getGLImplBaseClassName**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGLImplBaseClassName())() |  |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getGLTypeName**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getGLTypeName(int))​(int type) |  |
| [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getImpl**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getImpl())() | верните эту реализацию профилей, например. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getImplName**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getImplName())() | возвращает имя реализации этого профиля, например. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxFixedFunc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxFixedFunc(boolean))​(boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxFixedFunc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxFixedFunc(com.jogamp.nativewindow.AbstractGraphicsDevice,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, boolean favorHardwareRasterizer) | Возвращает самый высокий профиль, реализуя конвейер фиксированных функций. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaximum**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaximum(boolean))​(boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaximum**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaximum(com.jogamp.nativewindow.AbstractGraphicsDevice,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, boolean favorHardwareRasterizer) | Возвращает самый высокий профиль. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxProgrammable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxProgrammable(boolean))​(boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxProgrammable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxProgrammable(com.jogamp.nativewindow.AbstractGraphicsDevice,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, boolean favorHardwareRasterizer) | Возвращает самый высокий профиль, реализуя программируемый конвейер шейдеров. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxProgrammableCore**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxProgrammableCore(boolean))​(boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMaxProgrammableCore**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMaxProgrammableCore(com.jogamp.nativewindow.AbstractGraphicsDevice,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, boolean favorHardwareRasterizer) | Возвращает самый высокий профиль, реализуя *только* программируемый конвейер шейдерного *ядра*. |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMinimum**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMinimum(boolean))​(boolean favorHardwareRasterizer) | Использует устройство по умолчанию |
| static [**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) | [**getMinimum**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getMinimum(com.jogamp.nativewindow.AbstractGraphicsDevice,boolean))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, boolean favorHardwareRasterizer) | Возвращает самый низкий профиль. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**getName**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#getName())() | верните это имя профиля |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**glAvailabilityToString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#glAvailabilityToString())() | Использует устройство по умолчанию |
| static [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**glAvailabilityToString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#glAvailabilityToString(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) |  |
| static [**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) | [**glAvailabilityToString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#glAvailabilityToString(com.jogamp.nativewindow.AbstractGraphicsDevice,java.lang.StringBuilder))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, [**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) sb) |  |
| static [**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) | [**glAvailabilityToString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#glAvailabilityToString(com.jogamp.nativewindow.AbstractGraphicsDevice,java.lang.StringBuilder,java.lang.String,int))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, [**StringBuilder**](http://docs.oracle.com/javase/6/docs/api/java/lang/StringBuilder.html?is-external=true) sb, [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) indent, int indentCount) |  |
| boolean | [**hasGLSL**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#hasGLSL())() | Указывает, поддерживает ли данный профиль GLSL, т.е. |
| int | [**hashCode**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#hashCode())() |  |
| static void | [**initProfiles**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#initProfiles(com.jogamp.nativewindow.AbstractGraphicsDevice))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device) | Инициализируйте активную инициализацию GLProfiles для данного устройства, если это еще не сделано. |
| static void | [**initSingleton**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#initSingleton())() | Статическая инициализация JOGL. |
| static boolean | [**isAnyAvailable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isAnyAvailable())() | Возвращает доступность любого профиля на устройстве по умолчанию. |
| static boolean | [**isAvailable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isAvailable(com.jogamp.nativewindow.AbstractGraphicsDevice,java.lang.String))​([**AbstractGraphicsDevice**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/nativewindow/AbstractGraphicsDevice.html) device, [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profile) | Возвращает доступность профиля на устройстве. |
| static boolean | [**isAvailable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isAvailable(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profile) | Возвращает доступность профиля на устройстве по умолчанию. |
| static boolean | [**isAWTAvailable**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isAWTAvailable())() |  |
| boolean | [**isGL2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL2())() | Указывает , поддерживает ли этот профиль GL2 . |
| boolean | [**isGL2ES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL2ES1())() | Указывает, поддерживает ли этот профиль GL2ES1. |
| boolean | [**isGL2ES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL2ES2())() | Указывает, поддерживает ли этот профиль GL2ES2. |
| boolean | [**isGL2ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL2ES3())() | Указывает, поддерживает ли этот профиль GL2ES3. |
| boolean | [**isGL2GL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL2GL3())() | Указывает, поддерживает ли этот профиль GL2GL3. |
| boolean | [**isGL3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL3())() | Указывает, поддерживает ли этот профиль GL3. |
| boolean | [**isGL3bc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL3bc())() | Указывает, поддерживает ли этот профиль GL3bc. |
| boolean | [**isGL3ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL3ES3())() | Указывает, поддерживает ли этот профиль GL3ES3. |
| boolean | [**isGL4**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL4())() | Указывает, поддерживает ли этот профиль GL4. |
| boolean | [**isGL4bc**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL4bc())() | Указывает, поддерживает ли этот профиль GL4bc. |
| boolean | [**isGL4ES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGL4ES3())() | Указывает, поддерживает ли этот профиль GL4ES3. |
| boolean | [**isGLES**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGLES())() | Указывает, поддерживает ли этот профиль GLES. |
| boolean | [**isGLES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGLES1())() | Указывает, поддерживает ли этот профиль GLES1. |
| boolean | [**isGLES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGLES2())() | Указывает, поддерживает ли этот профиль GLES2. |
| boolean | [**isGLES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isGLES3())() | Указывает, поддерживает ли этот профиль GLES3. |
| boolean | [**isHardwareRasterizer**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isHardwareRasterizer())() | возвращает true, если impl. |
| static boolean | [**isInitialized**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isInitialized())() |  |
| boolean | [**isValidArrayDataType**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isValidArrayDataType(int,int,int,boolean,boolean))​(int index, int comps, int type, boolean isVertexAttribPointer, boolean throwException) |  |
| boolean | [**isValidDataType**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#isValidDataType(int,boolean))​(int type, boolean throwException) | Общая проверка, если тип является допустимым типом данных GL для текущего профиля |
| static void | [**shutdown**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#shutdown())() | Метод ручного завершения работы может быть вызван после последнего использования JOGL в запущенной JVM. Он освобождает все временно созданные ресурсы, то есть проблемы [GLDrawableFactory.shutdown()](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLDrawableFactory.html#shutdown()). Реализация завершения работы вызывается через перехват завершения работы JVM, если не вызывается вручную. |
| [**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [**toString**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#toString())() |  |
| boolean | [**usesNativeGLES**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES())() | Указывает, использует ли этот профиль любую из нативных реализаций OpenGL ES. |
| static boolean | [**usesNativeGLES**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profileImpl) | Указывает, используется ли какой-либо из нативных профилей OpenGL ES. |
| boolean | [**usesNativeGLES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES1())() | Указывает, использует ли этот профиль нативных реализации OpenGL ES1. |
| static boolean | [**usesNativeGLES1**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES1(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profileImpl) | Указывает, используется ли собственный профиль OpenGL ES1. |
| boolean | [**usesNativeGLES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES2())() | Указывает, использует ли этот профиль собственные реализации OpenGL ES2. |
| static boolean | [**usesNativeGLES2**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES2(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profileImpl) | Указывает, используется ли собственный профиль OpenGL ES3 или ES2. |
| boolean | [**usesNativeGLES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES3())() | Указывает, использует ли этот профиль собственные реализации OpenGL ES3. |
| static boolean | [**usesNativeGLES3**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#usesNativeGLES3(java.lang.String))​([**String**](http://docs.oracle.com/javase/6/docs/api/java/lang/String.html?is-external=true) profileImpl) | Указывает, используется ли собственный профиль OpenGL ES2. |
| void | [**verifyEquality**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html#verifyEquality(com.jogamp.opengl.GLProfile))​([**GLProfile**](https://jogamp.org/deployment/jogamp-next/javadoc/jogl/javadoc/com/jogamp/opengl/GLProfile.html) glp) |  |

Пример. Пытаемся понять профили, возможности и т.п.

**import** javax.swing.\*;

**import** **static** com.jogamp.opengl.GL2.\*;

**import** com.jogamp.opengl.\*;

**import** com.jogamp.opengl.awt.GLCanvas;

**public** **class** Code **extends** JFrame **implements** GLEventListener{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** GLCanvas myCanvas;

**public** Code(){

setTitle(" ");

setSize(600, 400);

setLocation(200, 200);

myCanvas = **new** GLCanvas();

//Свойства только что созданного GLCanvas

System.***out***.println("Свойства только что созданного GLCanvas с конструктором по умолчанию");

System.***out***.println(myCanvas);

myCanvas.addGLEventListener(**this**);

**this**.add(myCanvas);

setVisible(**true**);

System.***out***.println("Свойства GLCanvas, после того как к нему подключили слушателя");

System.***out***.println("и добавили его в окно JFrame");

System.***out***.println(myCanvas);

//System.out.println();

}

**public** **void** display(GLAutoDrawable drawable){

GL2 gl = (GL2) GLContext.*getCurrentGL*();

gl.glClearColor(0.1f, 0.5f, 0.5f, 0.5f);

gl.glClear(***GL\_COLOR\_BUFFER\_BIT***);

gl.glColor3f(1.0f, 0.0f, 0.0f);

gl.glRecti(0, 0, 100, 100);

gl.glFlush();

}

**public** **static** **void** main(String[ ] args){

**new** Code();

}

**public** **void** init(GLAutoDrawable drawable) {

// Профиль

GLProfile glp = GLProfile.*getDefault*();

System.***out***.println("Возможность использования профилей");

System.***out***.println(GLProfile.*glAvailabilityToString*());

System.***out***.println("Максимальный профиль");

System.***out***.println(GLProfile.*getMaximum*(**true**));

System.***out***.println("Профиль по умолчанию");

System.***out***.println(glp);

System.***out***.println("Аппаратная растеризация?");

System.***out***.println(glp.isHardwareRasterizer());

System.***out***.println("Имя профиля");

System.***out***.println(glp.getImplName());

// Возможности

GLCapabilities cap = **new** GLCapabilities(glp);

System.***out***.println("Возможности для профиля");

System.***out***.println(cap);

System.***out***.println("Возможности для профиля - установлена альфа 16 бит");

cap.setAlphaBits(16);

System.***out***.println(cap);

}

**public** **void** reshape(GLAutoDrawable drawable, **int** x, **int** y, **int** width, **int** height) { }

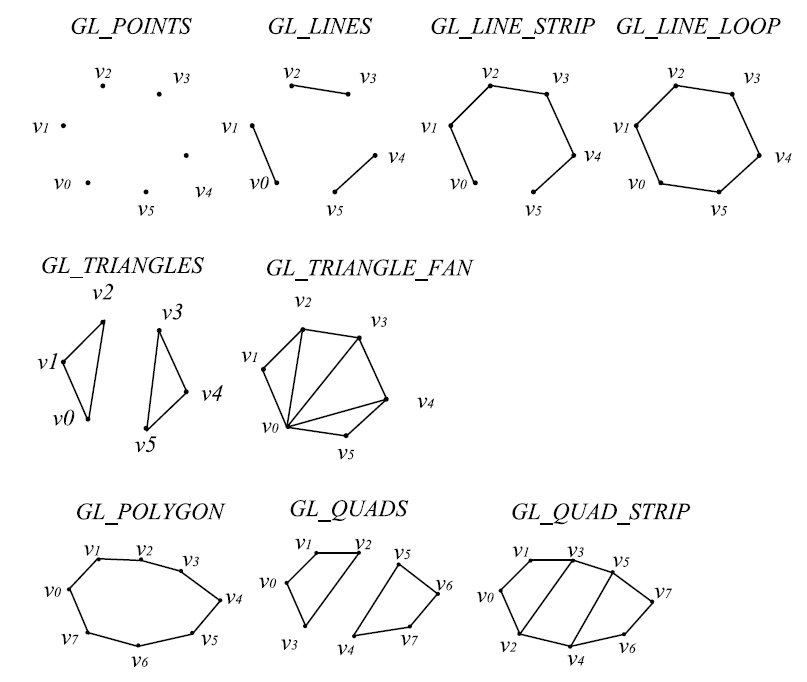
**public** **void** dispose(GLAutoDrawable drawable) { }

}

Двумерная графика на JOGL

Ниже приведены различные встроенные параметры OpenGL, поддерживаемые JOGL:

|  |  |
| --- | --- |
| **Примитивный** | **Описание** |
| GL\_POINT | Каждая вершина рассматривается как отдельная точка |
| GL\_LINES | Он обрабатывает каждую пару вершин как отрезок прямой. |
| GL\_LINES\_STRIP | Соединяет группу отрезков друг с другом. |
| GL\_LINES\_LOOP | Соединяет группу отрезков в цикле, т.е. от первого к последнему, а затем обратно к первому. |
| GL\_TRIANGLE | Каждая вершина триплета ведет себя как независимый треугольник |
| GL\_TRIANGLE\_FAN | Создает связную группу треугольников, где каждый треугольник определен для определенной вершины, представленной после первых двух вершин. |
| GL\_TRIANGLE\_STRIP | Соединяет группу треугольников друг с другом. |
| GL\_QUADS | Каждая группа из четырех вершин ведет себя как независимый четырехугольник. |
| GL\_QUAD\_STRIP | Соединяет группу четырехугольников друг с другом. |
| GL\_POLYGON | Рисует один выпуклый многоугольник. |



**Исследуем возможности**

**import** javax.swing.\*;

**import** **static** com.jogamp.opengl.GL2.\*;

**import** com.jogamp.opengl.\*;

**import** com.jogamp.opengl.awt.GLCanvas;

**public** **class** Code **extends** JFrame **implements** GLEventListener{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** GLCanvas myCanvas;

**public** Code(){

setTitle(" ");

setSize(600, 400);

setLocation(200, 200);

myCanvas = **new** GLCanvas();

//Свойства только что созданного GLCanvas

System.***out***.println("Свойства только что созданного GLCanvas с конструктором по умолчанию");

System.***out***.println(myCanvas);

myCanvas.addGLEventListener(**this**);

**this**.add(myCanvas);

setVisible(**true**);

System.***out***.println("Свойства GLCanvas, после того как к нему подключили слушателя");

System.***out***.println("и добавили его в окно JFrame");

System.***out***.println(myCanvas);

//System.out.println();

}

**public** **void** display(GLAutoDrawable drawable){

GL2 gl = (GL2) GLContext.*getCurrentGL*();

gl.glClearColor(0.1f, 0.5f, 0.5f, 0.5f);

gl.glClear(***GL\_COLOR\_BUFFER\_BIT***);

gl.glColor3f(1.0f, 0.0f, 0.0f);

gl.glRecti(0, 0, 100, 100);

gl.glFlush();

}

**public** **static** **void** main(String[ ] args){

**new** Code();

}

**public** **void** init(GLAutoDrawable drawable) {

// Профиль

GLProfile glp = GLProfile.*getDefault*();

System.***out***.println("Возможность использования профилей");

System.***out***.println(GLProfile.*glAvailabilityToString*());

System.***out***.println("Максимальный профиль");

System.***out***.println(GLProfile.*getMaximum*(**true**));

System.***out***.println("Профиль по умолчанию");

System.***out***.println(glp);

System.***out***.println("Аппаратная растеризация?");

System.***out***.println(glp.isHardwareRasterizer());

System.***out***.println("Имя профиля");

System.***out***.println(glp.getImplName());

// Возможности

GLCapabilities cap = **new** GLCapabilities(glp);

System.***out***.println("Возможности для профиля");

System.***out***.println(cap);

System.***out***.println("Возможности для профиля - установлена альфа 16 бит");

cap.setAlphaBits(16);

System.***out***.println(cap);

}

**public** **void** reshape(GLAutoDrawable drawable, **int** x, **int** y, **int** width, **int** height) { }

**public** **void** dispose(GLAutoDrawable drawable) { }

}

Пример. Рисуем двумерную линию

**import** javax.swing.JFrame;

**import** com.jogamp.opengl.GL2;

**import** com.jogamp.opengl.GLAutoDrawable;

**import** com.jogamp.opengl.GLCapabilities;

**import** com.jogamp.opengl.GLEventListener;

**import** com.jogamp.opengl.GLProfile;

**import** com.jogamp.opengl.awt.GLCanvas;

**public** **class** BasicLine **implements** GLEventListener {

@Override

**public** **void** init(GLAutoDrawable arg0)

{

}

@Override

**public** **void** display(GLAutoDrawable drawable) {

**final** GL2 gl = drawable.getGL().getGL2();

//Каждую пару точек трактуем как отдельную линию

gl.glBegin (GL2.***GL\_LINES***);

gl.glVertex2d(-0.60, 0.10);

gl.glVertex2d(0.60, 0.10);

gl.glVertex2d(0.10, -0.60);

gl.glVertex2d(0.10, 0.60);

gl.glEnd();

}

@Override

**public** **void** reshape(GLAutoDrawable arg0, **int** arg1, **int** arg2, **int** arg3, **int** arg4)

{

}

@Override

**public** **void** dispose(GLAutoDrawable arg0)

{

}

**public** **static** **void** main(String[] args) {

**final** GLProfile gp = GLProfile.*get*(GLProfile.***GL2***);

GLCapabilities cap = **new** GLCapabilities(gp);

**final** GLCanvas gc = **new** GLCanvas(cap);

BasicLine bl = **new** BasicLine();

gc.addGLEventListener(bl);

**final** JFrame frame = **new** JFrame ("Линия");

frame.add(gc);

frame.setSize(500,400);

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frame.setVisible(**true**);

}

}

**Рисуем простые трехмерные фигуры**

**import** java.awt.Dimension;

**import** java.awt.Frame;

**import** java.awt.Toolkit;

**import** java.awt.event.WindowEvent;

**import** java.awt.event.WindowListener;

**import** javax.swing.JFrame;

**import** com.jogamp.opengl.GL2;

**import** com.jogamp.opengl.GLAutoDrawable;

**import** com.jogamp.opengl.GLCapabilities;

**import** com.jogamp.opengl.GLEventListener;

**import** com.jogamp.opengl.GLProfile;

**import** com.jogamp.opengl.awt.GLCanvas;

**import** com.jogamp.opengl.glu.GLU;

**import** com.jogamp.opengl.util.gl2.GLUT;

**public** **class** JOGL3dDemo **implements** GLEventListener {

//GL2 gl;

**private** GLUT glut = **new** GLUT();

**private** GLU glu = **new** GLU();

**public** **static** **void** main(String[] args) {

// Получаем совместимость объекта с профилем GL2

**final** GLProfile profile = GLProfile.*get*(GLProfile.***GL2***);

GLCapabilities capabilities = **new** GLCapabilities(profile);

// Холст

**final** GLCanvas glcanvas = **new** GLCanvas(capabilities);

JOGL3dDemo b = **new** JOGL3dDemo();

glcanvas.addGLEventListener(b);

//Создание фрейма

**final** JFrame frame = **new** JFrame ("Трехмерные фигуры");

//System.out.println(frame);

//Добавление холста к фрейму

frame.add(glcanvas);

//Определяем размер фрейма так

//или так - размер фрейма равен размеру экрана

Dimension screenSize = Toolkit.*getDefaultToolkit*().getScreenSize();

frame.setSize(screenSize.width, screenSize.height);

glcanvas.setSize(screenSize.width, screenSize.height);

frame.addWindowListener(**new** WindowListener() {

@Override

**public** **void** windowActivated(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** windowClosed(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** windowClosing(WindowEvent arg0) {

System.*exit*(0);

}

@Override

**public** **void** windowDeactivated(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** windowDeiconified(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** windowIconified(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** windowOpened(WindowEvent arg0) {

// **TODO** Auto-generated method stub

}

});

frame.setVisible(**true**);

}

@Override

**public** **void** display(GLAutoDrawable drawable) {

GL2 gl = drawable.getGL().getGL2();

gl.glMatrixMode(GL2.***GL\_PROJECTION***);

gl.glLoadIdentity();

gl.glOrtho(-2, 2, -2, 2, 0.1, 100);

gl.glMatrixMode(GL2.***GL\_MODELVIEW***);

gl.glLoadIdentity();

glu.gluLookAt(1.0, 1.0, 2.0, 0, 0, 0, 0, 1, 0);

gl.glClear(GL2.***GL\_COLOR\_BUFFER\_BIT***);

gl.glColor3ub((**byte**)255, (**byte**)0, (**byte**)0);

// Ось z

**this**.axis(drawable, 0.5);

// ось y

gl.glPushMatrix();

gl.glRotated(90, 0, 1, 0);

**this**.axis(drawable, 0.5);

gl.glPopMatrix();

// ось x

//gl.glPushMatrix();

gl.glRotated(-90, 1, 0, 0);

**this**.axis(drawable, 0.5);

gl.glPopMatrix();

// Куб

gl.glPushMatrix();

gl.glTranslated(0.7, 0.7, 0.7);

glut.glutWireCube((**float**)1.0);

gl.glPopMatrix();

// Сфера

gl.glPushMatrix();

gl.glTranslated(1.0, 1.0, 0);

glut.glutWireSphere(0.2, 20, 18);

gl.glPopMatrix();

// Конус

gl.glPushMatrix();

gl.glTranslated(1.0, 0, 1.0);

glut.glutWireCone(0.2, 0.5, 10, 8);

gl.glPopMatrix();

// Чайник

gl.glPushMatrix();

gl.glRotated(90, 90, 90, 0);

gl.glTranslated(-0.5, -0.5, -0.5);

glut.glutWireTeapot(0.2);

gl.glPopMatrix();

}

@Override

**public** **void** dispose(GLAutoDrawable arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** init(GLAutoDrawable arg0) {

// **TODO** Auto-generated method stub

}

@Override

**public** **void** reshape(GLAutoDrawable arg0, **int** arg1, **int** arg2, **int** arg3, **int** arg4) {

// **TODO** Auto-generated method stub

}

// Римсем ось

**void** axis (GLAutoDrawable drawable, **double** lenght) {

**final** GL2 gl = drawable.getGL().getGL2();

gl.glPushMatrix();

gl.glBegin(GL2.***GL\_LINES***);

gl.glVertex3d(0, 0, 0);

gl.glVertex3d(0, 0, lenght);

gl.glEnd();

gl.glTranslated(0, 0, lenght - 0.2);

glut.glutWireCone(0.04, 0.2, 12, 9);

gl.glPopMatrix();

}

}

**Варианты заданий**

