Actor	Actor methods are available to all actors.
World	World methods are available to the world.
Greenfoot	Used to communicate with the environment itself.
GreenfootImage	For presenting and manipulating images.

Class Actor

`Actor()`	Construct an Actor.
`void act()`	The act method is called by the greenfoot framework to give objects a chance to perform some action.
`protected void addedToWorld(World world)`	This method will be called by the Greenfoot system when the object has been inserted into the world.
`int getHeight()`	Return the height of the object.
`GreenfootImage getImage()`	Returns the image used to represent this Actor.
`protected List getIntersectingObjects (Class cls)`	Return all the objects that intersect this object.
`protected List getNeighbours (int distance, boolean diagonal, Class cls)`	Return the neighbours to this object within a given distance.
`protected List getObjectsAtOffset(int dx, int dy, Class cls)`	Return all objects that intersect the given location (relative to this object's location).
`protected List getObjectsInRange(int r, Class cls)`	Return all objects within range 'r' around this object.
`protected Actor getOneIntersectingObject (Class cls)`	Return an object that intersects this object.
`protected Actor getOneObjectAtOffset (int dx, int dy, Class cls)`	Return one object that is located at the specified cell (relative to this objects location).
`int getRotation()`	Return the current rotation of the object.
`int getWidth()`	Return the width of the object.
`World getWorld()`	Return the world that this object lives in.
`int getX()`	Return the x-coordinate of the object's current location.
`int getY()`	Return the y-coordinate of the object's current location.
`protected boolean intersects(Actor other)`	Check whether this object intersects another given object.
`void setImage(GreenfootImage image)`	Set the image for this object to the specified image.
`void setImage(String filename)`	Set an image for this object from an image file.
`void setLocation(int x, int y)`	Assign a new location for this object.
`void setRotation(int rotation)`	Set the rotation of the object.

Class World	
World(int worldWidth, int worldHeight, int cellSize)	Construct a new world.
void **addObject**(Actor object, int x, int y)	Add an Actor to the world (at the object's specified location).
GreenfootImage **getBackground**()	Return the world's background image.
int **getCellSize**()	Return the size of a cell (in pixels).
Color **getColorAt**(int x, int y)	Return the color at the center of the cell.
int **getHeight**()	Return the height of the world (in number of cells).
List **getObjects**(Class cls)	Get all the objects in the world. If iterating through these objects, you should synchronize on this world to avoid ConcurrentModificationException.
List **getObjectsAt**(int x, int y, Class cls)	Return all objects at a given cell.
int **getWidth**()	Return the width of the world (in number of cells).
boolean **isTiled**()	Returns true if the world is tiled.
void **removeObject**(Actor object)	Remove an object from the world.
void **removeObjects**(Collection objects)	Remove a list of objects from the world.
void **repaint**()	Repaint the world.
void **setBackground**(GreenfootImage image)	Set a background image for the world.
void **setBackground**(String filename)	Set a background image for the world from an image file.
void **setTiled**(boolean tiled)	If set to true, the background image will be tiled to fill out the entire background of the world.

Class Greenfoot	
Greenfoot()	Constructor.
static void **delay**()	Delay the current execution by the time specified by the Greenfoot environment (the speed slider).
static String **getKey**()	Get the most recently pressed key, since the last time this method was called.
static int **getRandomNumber**(int limit)	Return a random number between 0 (inclusive) and limit (exclusive).
static boolean **isKeyDown**(String keyName)	Check whether a given key is currently pressed down.
static void **playSound**(String soundFile)	Play sound from a file.
static void **setSimulationSpeed**(int speed)	Set the speed of the simulation execution.
static void **startSimulation**()	Run (or resume) the simulation.
static void **stopSimulation**()	Stop the simulation.

Class GreenfootImage	
GreenfootImage(GreenfootImage image)	Create a GreenfootImage from another GreenfootImage.
GreenfootImage(int width, int height)	Create an empty (transparent) image with the specified size.
GreenfootImage(String filename)	Create an image from an image file.
void **clear**()	Clear the image.
void **drawImage**(GreenfootImage image, int x, int y)	Draws the given Image onto this image.
void **drawLine**(int x1, int y1, int x2, int y2)	Draw a line, using the current drawing color, between the points (x1, y1) and (x2, y2).
void **drawOval**(int x, int y, int width, int height)	Draw an oval bounded by the specified rectangle with the current drawing color.
void **drawPolygon**(int[] xPoints, int [] yPoints, int nPoints)	Draws a closed polygon defined by arrays of x and y coordinates.
void **drawRect**(int x, int y, int width, int height)	Draw the outline of the specified rectangle.
void **drawString**(String string, int x, int y)	Draw the text given by the specified string, using the current font and color.
void **fill**()	Fill the entire image with the current drawing color.
void **fillOval**(int x, int y, int width, int height)	Fill an oval bounded by the specified rectangle with the current drawing color.
void **fillPolygon**(int[] xPoints, int[] yPoints, int nPoints)	Fill a closed polygon defined by arrays of x and y coordinates.
void **fillRect**(int x, int y, int width, int height)	Fill the specified rectangle.
Color **getColor**()	Return the current drawing color.
Color **getColorAt**(int x, int y)	Return the color at the given pixel.
Font **getFont**()	Get the current font.
int **getHeight**()	Return the height of the image.
int **getWidth**()	Return the width of the image.
void **mirrorHorizontally**()	Mirror the image horizontally (flip around the x-axis).
void **mirrorVertically**()	Mirror the image vertically (flip around the y-axis).
void **rotate**(int degrees)	Rotates this image around the center.
void **scale**(int width, int height)	Scales this image to a new size.
void **setColor**(Color color)	Set the current drawing color.
void **setColorAt**(int x, int y, Color color)	Sets the color at the given pixel to the given color.
void **setFont**(Font f)	Set the current font.