JavaScript Game Development Library

# Sprites and SpriteSheets

There are 3 different types of image classes in this library. One for regular Sprites (single image) and two for SpriteSheets. The two SpriteSheets differ only in whether the distinct “frames” come from one file or multiple files.

# Sprite

Use this class is you have an image with no frames (so no animation). Especially good for backgrounds, terrain, walls, etc.

### Properties (“this.property”)

|  |  |
| --- | --- |
| x | X position where image is to be painted on the canvas |
| y | Y position where image is to be painted on the canvas |
| width | Actual (native) width of the image in pixels |
| height | Actual (native) height of the image in pixels |
| useWidth | Width to use when painting the image. Image will be stretched/shrunk if different from the actual (native) width. |
| useHeight | Height to use when painting the image. Image will be stretched/shrunk if different from the actual (native) height. |
| dX | Change in direction in the X axis for each screen repaint (if used in calculations) |
| dY | Change in direction in the Y axis for each screen repaint (if used in calculations) |
| image | The image variable for the image. Note that all images must first be loaded into image variables before using in these classes. |
| alpha | Transparency of the Spite from 0.0 (invisible) to 1.0 (fully visible) |
| collision | Indicates if the Sprite is in collision with another object or not. This is only checked when one of the collision methods are called. Note that every time a collision method is called, collision is first set to false and then checked for collision again. This is important, since the criteria for the different collision methods can be different. |
| visible | Indicates if the image is to be painted or not when draw is called. Can also be used to decide if image should be used for collision detection. |
| moveWithBackground | Will cause the sprite image to move along with the background if set to true. Note that if you also have the Sprite move on its own the Sprite will move a combination of what YOU told it to move and what the background moved. |

### Methods (“this.method()”)

|  |  |
| --- | --- |
| draw() | Using all the current properties, the image is drawn on the canvas |
| checkCollisions(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) are in collision or not. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkBottomCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves DOWN one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkTopCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves UP one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkRightCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves RIGHT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkLeftCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves LEFT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |

### Usage:

var background = new Image();

background.src = game.imageDir + "maze2.png";

var bg = new Sprite(0,0,1200,600,1200,600, background);

1. Create a new image variable.
2. Then set the image file (src) to your image’s file name. Note that if the image is not in the same folder as your HTML, you’ll need to provide the path to the image as well in the name. Most likely, you will put your images in the “images” folder in your main game folder.
3. Create the Sprite variable passing the following values in order:
   1. X position to paint
   2. Y position to paint
   3. Native image width in pixels
   4. Native image height in pixels
   5. Width to use in pixels
   6. Height to use in pixels
   7. Image variable from step #1

# SpriteSheet

Use this for an image that has animation/motion. Use if all the “frames” are collected within a single image file.

### Properties (“this.property”)

|  |  |
| --- | --- |
| x | X position where image is to be painted on the canvas |
| y | Y position where image is to be painted on the canvas |
| width | Actual (native) width of the image in pixels |
| height | Actual (native) height of the image in pixels |
| useWidth | Width to use when painting the image. Image will be stretched/shrunk if different from the actual (native) width. |
| useHeight | Height to use when painting the image. Image will be stretched/shrunk if different from the actual (native) height. |
| dX | Change in direction in the X axis for each screen repaint (if used in calculations) |
| dY | Change in direction in the Y axis for each screen repaint (if used in calculations) |
| image | The image variable for the image. Note that all images must first be loaded into image variables before using in these classes. |
| alpha | Transparency of the Spite from 0.0 (invisible) to 1.0 (fully visible) |
| collision | Indicates if the Sprite is in collision with another object or not. This is only checked when one of the collision methods are called. Note that every time a collision method is called, collision is first set to false and then checked for collision again. This is important since the criteria for the different collision methods can be different. |
| visible | Indicates if the image is to be painted or not when draw is called. Can also be used to decide if image should be used for collision detection. |
| moveWithBackground | Will cause the sprite image to move along with the background if set to true. Note that if you also have the Sprite move on its own the Sprite will move a combination of what YOU told it to move and what the background moved. |
| dirMax | Max number of "directions" the Sprite can move in |
| currentDir | The current “row” or direction the Sprite is drawing (use array numbering) |
| maxFrame | Maximum number of frames (columns) in the Sprite Sheet |
| currentFrame | Current Frame (column) of the Sprite Sheet being drawn (use array numbering) |
| canJump \*\*\* | Indicates if the Sprite can jump \*\*\* |
| gravity \*\*\* | Indicates how fast the Sprite will fall – simulates gravity \*\*\* |
| isJumping \*\*\* | Indicates it the Sprite is currently jumping \*\*\* |
| jumpMax \*\*\* | How high the Sprite can jump before it should start falling again \*\*\* |
| jumpCount \*\*\* | How many pixels the Sprite has jumped thus far. Helps determine when it should start falling again. \*\*\* |

### Methods (“this.method()”)

|  |  |
| --- | --- |
| draw() | Using all the current properties, the image is drawn on the canvas |
| advanceFrame() | Will move to the next Frame (next image in column) in this SpriteSheet or move back to first Frame if currently on the last Frame |
| changeDir(dir) | Will change to specified row (dir is a number) of the SpriteSheet |
| checkCollisions(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) are in collision or not. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkBottomCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves DOWN one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkTopCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves UP one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkRightCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves RIGHT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkLeftCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves LEFT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| jump() | Starts a jump – if allowed |
| checkJump() | Use if Sprite may jump – will check if is in the process of jumping and will adjust Sprite y value if remaining jump count > 0 – else it will stop the jump |
| applyGravity() | Will apply gravity to Sprite ( y value) need to check on your own if this is appropriate ( i.e. if not resting on something that should stop it from falling. |

### Usage

var img = new Image();

img.src = game.imageDir + "pacman.png";

var pacman = new SpriteSheet(100,100,128,128,32,32,3,3,img);

1. Create a new image variable.
2. Then set the image file (src) to your image’s file name. Note that if the image is not in the same folder as your HTML, you’ll need to provide the path to the image as well in the name.
3. Create the Sprite variable passing the following values in order:
   1. X position to paint
   2. Y position to paint
   3. Native frame image width in pixels (take native width and divide by columns)
   4. Native frame image height in pixels (take native height and divide by rows)
   5. Width to use in pixels
   6. Height to use in pixels
   7. Direction – the number of rows for the 2-dimensional array of images
   8. Frame – the number of columns for the 2-dimensional array of images
   9. Image variable from step #1

# SpriteArray

Use this for an image that has animation/motion. Use if all the “frames” are spread across multiple image files. You will first need to compile all the images into an array of images.

### Properties (“this.property”)

|  |  |
| --- | --- |
| x | X position where image is to be painted on the canvas |
| y | Y position where image is to be painted on the canvas |
| width | Actual (native) width of the image in pixels |
| height | Actual (native) height of the image in pixels |
| useWidth | Width to use when painting the image. Image will be stretched/shrunk if different from the actual (native) width. |
| useHeight | Height to use when painting the image. Image will be stretched/shrunk if different from the actual (native) height. |
| dX | Change in direction in the X axis for each screen repaint (if used in calculations) |
| dY | Change in direction in the Y axis for each screen repaint (if used in calculations) |
| image | The image variable for the image. Note that all images must first be loaded into image variables before using in these classes. |
| alpha | Transparency of the Spite from 0.0 (invisible) to 1.0 (fully visible) |
| collision | Indicates if the Sprite is in collision with another object or not. This is only checked when one of the collision methods are called. Note that every time a collision method is called collision is first set to false and then checked for collision again. This is important since the criteria for the different collision methods can be different. |
| visible | Indicates if the image is to be painted or not when draw is called. Can also be used to decide if image should be used for collision detection. |

### Methods (“this.method()”)

|  |  |
| --- | --- |
| draw() | Using all the current properties, the image is drawn on the canvas |
| checkCollisions(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) are in collision or not. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkBottomCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves DOWN one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkTopCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves UP one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkRightCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves RIGHT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |
| checkLeftCollision(obj) | Returns a Boolean indicating if this Sprite and the passed object (can be another Sprite, SpriteSheet, or SpriteArray) WILL BE IN COLLISION if this Sprite moves LEFT one more pixel. True = they are; false = they are not.  It also sets the Sprites collision property to the appropriate value as well. |

### Usage

var starArray = [];

starArray[0] = game.imageDir + "star coin rotate 1.png";

starArray[1] = game.imageDir + "star coin rotate 2.png";

starArray[2] = game.imageDir + "star coin rotate 3.png";

starArray[3] = game.imageDir + "star coin rotate 4.png";

starArray[4] = game.imageDir + "star coin rotate 5.png";

starArray[5] = game.imageDir + "star coin rotate 6.png";

var aStar = new SpriteArray(45,45,1957,2242,45,45,1,2,starArray);

1. Create the array that will store all the images.
2. Load the images into individual elements within the array (images in array should be same size)
3. Create the Sprite variable passing the following values in order:
   1. X position to paint
   2. Y position to paint
   3. Native image width in pixels (all images in array should be same size)
   4. Image height in pixels (all images in array should be same size)
   5. Width to use in pixels
   6. Height to use in pixels
   7. Direction – the number of rows for the 2-dimensional array of images
   8. Frame – the number of columns for the 2-dimensional array of images
   9. Image array from step #1

# ScrollingBackGround

This Object is very much like a regular Sprite, except special features have been added to enable scrolling of the background image.

### Properties (“this.property”)

|  |  |
| --- | --- |
| x | X position where image is to be painted on the canvas |
| Y | Y position where image is to be painted on the canvas |
| width | Actual width of the image in pixels |
| height | Actual height of the image in pixels |
| useWidth | Width to use when painting the image. Image will be stretched/shrunk if different from the actual width. |
| useHeight | Height to use when painting the image. Image will be stretched/shrunk if different from the actual height. |
| dX | Change in direction in the X axis for each screen repaint |
| dY | Change in direction in the Y axis for each screen repaint |
| image | The image variable for the image. Note that all images must first be loaded into image variables before using in these classes. |
| alpha | Transparency of the Spite from 0.0 (invisible) to 1.0 (fully visible) |
| collision | Indicates if the Sprite is in collision with another object or not. This is only checked when one of the collision methods are called. Note that every time a collision method is called collision is first set to false and then checked for collision again. This is important since the criteria for the different collision methods can be different. |
| visible | Indicates if the image is to be painted or not when draw is called. Can also be used to decide if image should be used for collision detection. |
| advanceBackground | Boolean that indicates IF the background is to scroll. Utilizes the bgDX and bgDY values from the GameMaster object to perform this action |

### Methods (“this.method()”)

|  |  |
| --- | --- |
| draw() | Using all the current properties, the image is drawn on the canvas |

### Usage

var background = new Image();

background.src = game.imageDir + "maze2.png";

var bg = new ScrollingBackGround(0,0,1200,600,1200,600,background);

1. Load the images into individual image variable
2. Then set the image file (src) to your image’s file name. Note that if the image is not in the same folder as your HTML, you’ll need to provide the path to the image as well in the name.
3. Create the Sprite variable passing the following values in order:
4. X position to paint
5. Y position to paint
6. Native image width in pixels
7. Native image height in pixels
8. Width to use in pixels
9. Height to use in pixels
10. Image array #1

# GameText

Lets you print text to the canvas – like a Score.

### Properties (“this.property”)

|  |  |
| --- | --- |
| x | X position where image is to be painted on the canvas |
| y | Y position where image is to be painted on the canvas |
| font | Font to use – note must be on the computer |
| fillStyle | Color |
| alpha | Transparency of the GameText from 0.0 (invisible) to 1.0 (fully visible) |

### Methods (“this.method()”)

|  |  |
| --- | --- |
| draw() | Using all the current properties, the image is drawn on the canvas |

### Usage

1. var score = new GameText(); //create the text object
2. Optionally change the font, fillStyle, alpha and x, & y coordinates of score
3. score.draw("SCORE: 1200"); //pass in parenthesis what you want to print

# Keyboard mapping:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Code | Name | Code | Name | Code | Name | Code | Name |
| 8 | backSpace | 54 | key\_6 | 86 | key\_v | 114 | f3 |
| 9 | tab | 55 | key\_7 | 87 | key\_w | 115 | f4 |
| 13 | enter | 56 | key\_8 | 88 | key\_x | 116 | f5 |
| 16 | shift | 57 | key\_9 | 89 | key\_y | 117 | f6 |
| 17 | ctrl | 65 | key\_a | 90 | key\_z | 118 | f7 |
| 18 | alt | 66 | key\_b | 91 | leftWindowKey | 119 | f8 |
| 19 | pauseBreak | 67 | key\_c | 92 | rightWindowKey | 120 | f9 |
| 20 | capsLock | 68 | key\_d | 93 | selectKey | 121 | f10 |
| 27 | escape | 69 | key\_e | 96 | numpad0 | 122 | f11 |
| 33 | pageUp | 70 | key\_f | 97 | numpad1 | 123 | f12 |
| 34 | pageDown | 71 | key\_g | 98 | numpad2 | 144 | numLock |
| 35 | end | 72 | key\_h | 99 | numpad3 | 145 | scrollLock |
| 36 | home | 73 | key\_i | 100 | numpad4 | 186 | semiColon |
| 37 | leftArrow | 74 | key\_j | 101 | numpad5 | 187 | equalSign |
| 38 | upArrow | 75 | key\_k | 102 | numpad6 | 188 | comma |
| 39 | rightArrow | 76 | key\_l | 103 | numpad7 | 189 | dash |
| 40 | downArrow | 77 | key\_m | 104 | numpad8 | 190 | period |
| 45 | insert | 78 | key\_n | 105 | numpad9 | 191 | forwardSlash |
| 46 | delete | 79 | key\_o | 106 | multiplyKey | 192 | graveAccent |
| 48 | key\_0 | 80 | key\_p | 107 | addKey | 219 | openBracket |
| 49 | key\_1 | 81 | key\_q | 109 | subtractKey | 220 | backSlash |
| 50 | key\_2 | 82 | key\_r | 110 | decimalPoint | 221 | closeBracket |
| 51 | key\_3 | 83 | key\_s | 111 | divideKey | 222 | singleQuote |
| 52 | key\_4 | 84 | key\_t | 112 | f1 |  |  |
| 53 | key\_5 | 85 | key\_u | 113 | f2 |  |  |