Scratching Stage.java

Functions and Properties reference

We make a Stage object by *declaring* it at the top of our code, just like a *variable* or *Sprite.*
While we might have many Stage objects (and *child classes*), we should only have one Stage in each Sketch.

Example:

```
Stage theStage = new Stage(this);    // creates the Stage object
```

Properties are accessed a command such as

```
myVariable = theStage.property;
```

int Stage.backdropNumber;

This holds the number of the current backdrop.

int Stage.numberOfBackdrops;

This holds the number of backdrops the Stage has

Functions and Procedures

A *void* function is called simply, such as

myStage.move(10);   // move the Stage 10 pixels in the current direction

Others, such as *int* functions, return *calculated values*, much like a *variable*.

int myVariable = myStage.timer(); // store the time elapsed in seconds

void Stage.update();

Takes no arguments. This performs the actual rendering of the Stage.
It is necessary to call Stage.update() once per draw() loop, and should be just about the first thing we do in the draw() loop.

Example call:
myStage.update(); // display Stage on stage during the current loop

void loadDefaultBackdrop();

This is called automatically by the initializer, and loads the Scratch xy-grid backdrop.
You can modify this to replace the image with your own art.

Example call:
You should never have to call this function.
loadDefaultBackdrops() is called by the initializer when you declare a new Stage object.

void addBackdrop(string filepath);

This adds a backdrop to the Stage from an image in your Sketch folder, specified by the "file path" string. This file **must** be 480 x 360. Files should be carefully sorted in your Sketch folder.
Documents/Processing/mySketch/art/background/title_screen.png
Documents/Processing/mySketch/art/background/game_over_screen.png
Documents/Processing/mySketch/art/background/level1_screen.png
Documents/Processing/mySketch/art/background/level2_screen.png
Et cetera.

After you draw (or download and edit) and place the files, add them to your Stage with the lines
addBackdrop("art/background/title_screen.png");
addBackdrop("art/background/game_over_screen.png");
et cetera

Note that you **do not include the full path to your Sketch**, only the path to the image file.

void nextBackdrop();
void previousBackdrop();

Changes to the next or previous backdrop.
If you are on the first or last backdrop, these will intelligently loop to the opposite.

Example call:
myStage.nextBackdrop();
myStage.previousBackdrop();

void setBackdrop(int newBackdropNumber);

Change to a specific backdrop. This means remembering backdrop numbers.
Or you can add *constant values* which give those numbers easy names to remember.
Define them as follows

```
public static int title_screen=0;          // static means the value will not change
public static int game_over_screen=1;      // static means the value will not change
```

Example call:

```
myStage.switchToBackdrop(3);
myStage.switchToBackdrop(myStage.game_over_screen);
```

void show();
void hide();

These functions Show and Hide a Stage, like the Scratch blocks. These are provided to make adapting existing projects more straightforward. You could just as easily access the boolean *Stage.visible* to set visibility.

Example call:

```
myStage.show();
myStage.hide();
```

void resetTimer();
int timer();

The timer will start counting up immediately when you Sketch is run.
Use the resetTimer() function to reset.
Use the timer() function to return the number of seconds elapsed since the start (or last resetTimer(); call)

Example call:

```
myStage.resetTimer();                      // reset time to 0
int timeSinceReset = myStage.timer();      // read time since last reset
```

void ask(String yourQuestion);
String answer();

Processing has some built-in functions for handling simple keyboard input. To take longer input such as full words or sentences, as the Scratch “ask” and “answer” blocks do, we’ll need

Example call:

```
myStage.ask(“What is your quest?”);        // begin asking the question; only needs one call
myStage.update();                          // the question box is drawn by stage.update()
String someString = myStage.answer();      // store the answer in someString.
                                           // answer() returns an empty string if user is typing
```

void questionKeycheck();

In a sketch that asks questions, this function **must** be called inside your sketch in the **void keyPressed()** function, show below. ***If you do not call questionKeycheck correctly, the myStage.ask(yourQuestion) and myStage.answer() functions will not work.***

Example call:

```
void keyPressed() {
  myStage.questionKeycheck();
 // other keyboard logic may go here
}
```