



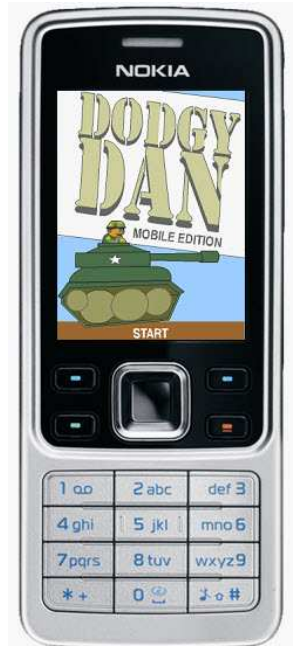
MAKE A DODGE-EM GAME FOR YOUR MOBILE

In this tutorial, you will learn how to create your first mobile game using Flash CS4 and Adobe Device Central CS4.

The game you create will be a simple dodge-em style game where the player must use the left and right keys on the phone to help their character dodge falling objects. The game will end in two ways.

The first way is if the player gets hit by too many of the falling objects. In this case, they will be taken to a game over scene where they will be told they failed. The second way is if the player manages to help the character dodge the objects without losing all of their lives for thirty seconds. In this case, they will get a 'you did it' style message to tell them of their success.

The game you create will be just like my Dodgy Dan Mobile Edition Game that can be played and downloaded from the Flash Classroom site at www.flashclassroom.com.



CHOOSING YOUR TARGET MOBILE DEVICE

Before you start planning and building your game, you will need to select the mobile device you will design your game for. If you have a mobile phone that runs Flash Lite Player 2.0 or above, you can create your game for use on your own mobile. If not, you may want to develop your game for use on one of your friend's mobiles or even just use the mobile device I have chosen - the Nokia 6290.

You will be able to check if your mobile phone supports Flash Lite Player 2.0 when we use Adobe Device Central in one of the following steps. Adobe Device Central comes with Flash CS4 and is very easy to use. It is an application that contains device profiles and emulators of all the mobile devices that support (or run) the Flash Lite player. The Flash Lite Player is the Flash Player that works on mobile devices. It is a cut down version of the normal Flash Player that displays Flash content on the web. Emulators are 'virtual versions' of the mobile device. They allow you to test the game or application you are building on your computer.

Let's begin by creating a new file and choosing our device.

1. Open up Flash CS4 and select **Create New > Flash File (Mobile)**.



2. When you select this option, it opens up **Adobe Device Central CS4**. At the top of Device Central you will see a tab called **New Document**. In this tab, change the Player Version to **Flash Lite 2.0**. Leave the other options as is.





FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

3. In this step, you will download the device profiles and emulators for the mobile phone you want to create your game for.

In the **Online Library** panel in the bottom left of the screen, you will see a list of mobile phone providers.



Click on your provider and then select **Devices > Download to Local Library** from the main menu at the top of Device Central.

4. Once the devices are downloaded from the Online Library, you will find the mobile phone provider in the **Local Library** (above the online library).

Select the name of your provider and you will see the compatible devices appear in the centre frame.



5. **Select your mobile phone** if it is there. If not, choose another mobile device to create your game for.

Click on your device to highlight it and then select the **Create** button in the bottom right hand corner of device central.



When you click this button, Flash opens up with a new file that is the correct size for the mobile device you have chosen.

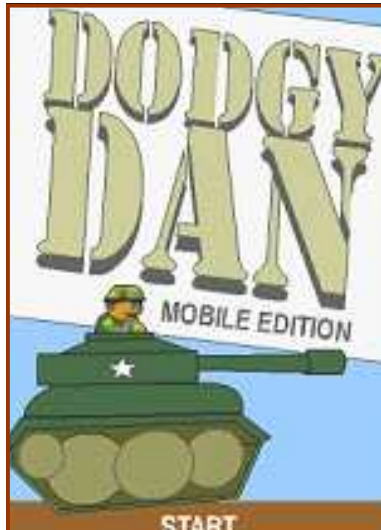
6. To see how Flash CS4 and Adobe Device Central CS4 work together to help you build your mobile game, draw a shape on the stage and then select **Control > Test Movie**. This will launch your file in the emulator for your device in Adobe Device Central. Each time you test your game, it will launch in this emulator and you will be able to use the buttons on the emulator as you would on the real device.
7. Click back on Adobe Flash and select **File > Save**. Save your file for use later.



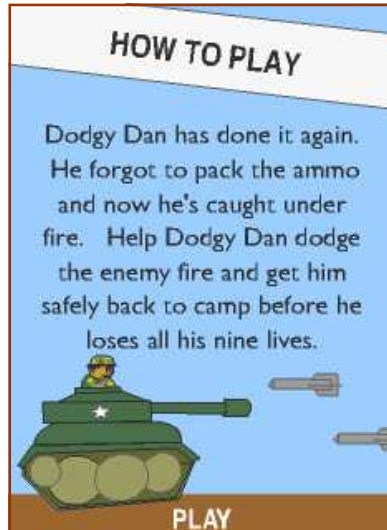


PLANNING YOUR GAME

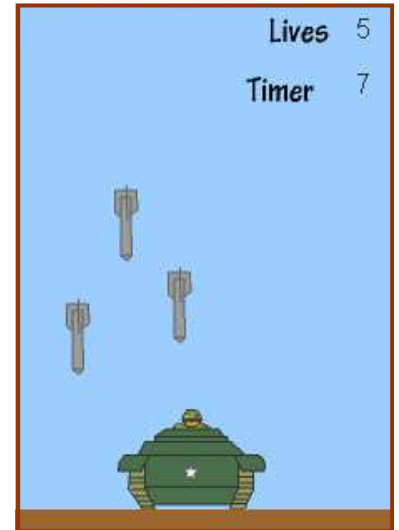
Before we start building the game, we need to do some planning. Your game will have five different scenes. An overview of each scene is provided below. These overviews are accompanied by screenshots from each scene in my demo game 'Dodgy Dan'.



Title Scene- This scene contains the title artwork and text. It also contains the word 'Start' above the enter key on the phone.



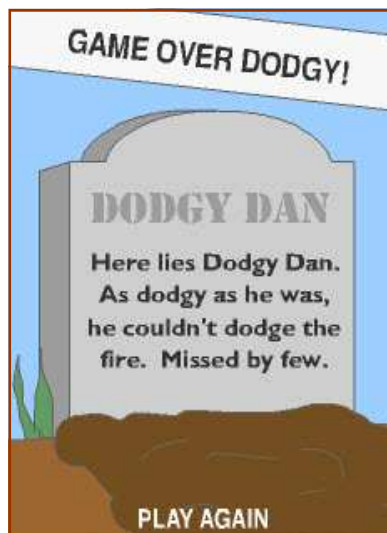
Instructions scene - This scene sets up the context for the game. It contains the text that explains the game play.



Game scene - This scene contains the game play. It includes the character and the objects that must be avoided. It also contains a timer and a counter for the number of lives.



You Did It Scene- This is the scene the player makes it to if they get through the game. It contains some sort of well done message.



Game Over Scene- This is the scene the player is taken to if they are hit by too many of the falling objects.

- It is now time for you to plan and develop a storyboard for each scene in your game. This tutorial will only show you how to create a game with the same features as those shown above so make sure your game has only these features. You can of course be creative and change the artwork, character and objects.

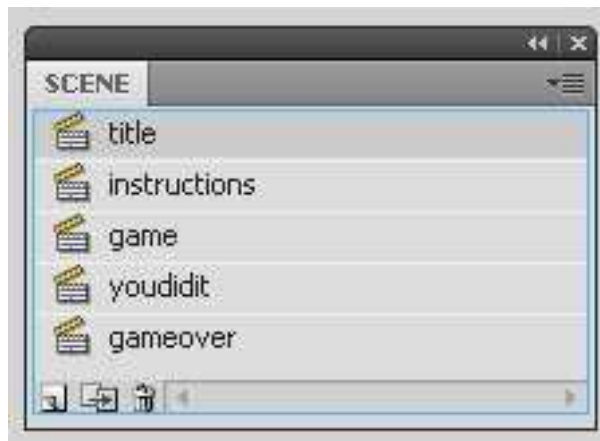
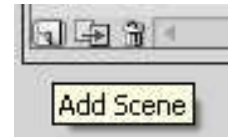
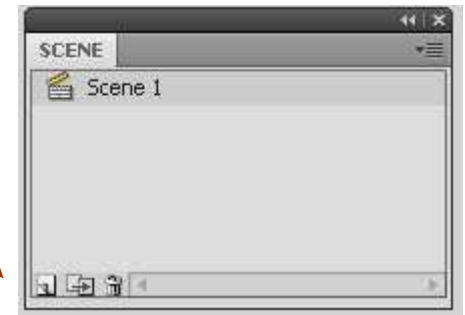




SETTING UP THE SCENES FOR YOUR GAME

It's time to set up the scenes for your game. We will create empty scenes to begin with and then work through the creation of each.

9. To set up the scenes for your game, **select Window > Other Panels > Scene.**
10. Create four additional scenes by selecting the **Add Scene** button **four times.**
11. Double click on the name of each scene in the panel and **rename** them so that they appear as below. Ensure there are no spaces between words.



Your scenes are now set up and ready to be edited and turned into the game for your mobile device.

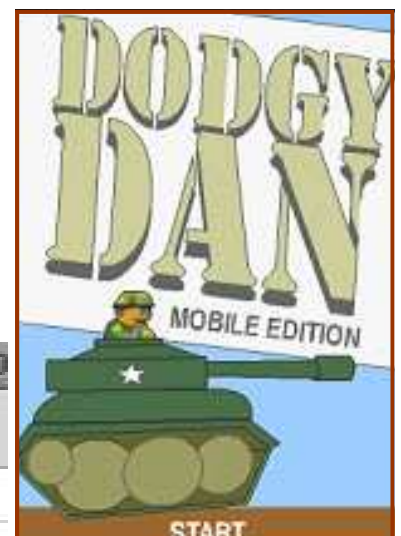
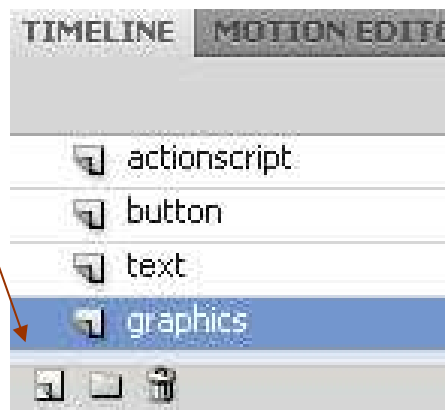
CREATING THE TITLE SCENE

This scene contains the title artwork and text. It also contains the word 'Start' above the enter key on the phone.

12. To edit this scene, click on the **Title** scene in the Scenes Panel.
13. We will now set up the **layers** for you to create the graphics, text, button and actionscript for this scene on.

To add new layers, click on the **New Layer** button.

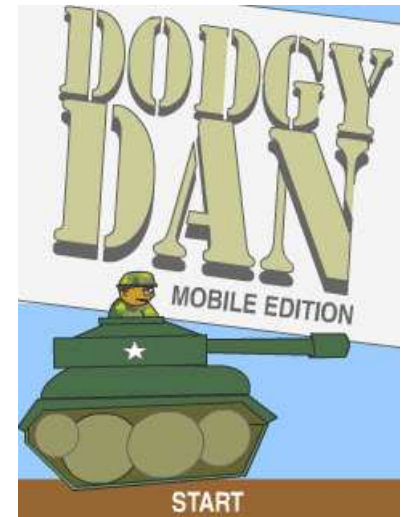
Rename each layer by double clicking on each layer name and typing in the text. Your layers should be named as shown.



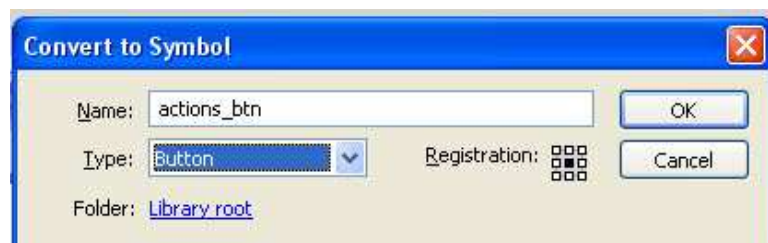


FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

14. Select the **graphics** layer and **draw the images** you want in your title scene.
15. Select the **text** layer and **add the text** for your title scene. This should include the word **Start** in the bottom centre of the design (as shown here).
16. Select the **button** layer and draw a circle to the right of the stage (as shown below).



17. Select the circle and press **F8** to convert the circle to a symbol. Give the symbol the name **actions_btn** and choose the **Button** type.



18. We will now add some actionscript to this button to allow the player to be taken to the instructions scene when they click on the Enter button.

To do this, **select on the actions_btn** to the right of your stage. **Press F9 to open the Actions Panel.**

In the actionscript panel, **type in the following code.**



19. It is time to add the final item in our Title Scene. This is a small bit of script that will tell the Flash Lite Player to stop at this scene until the player presses the Enter key on the device.

Click on Frame 1 of the Actionscript Layer.
Press **F9** to once again open the **Actions Layer**.
Add the `stop();` action as shown here. →



20. You have now completed the Title Scene. To **test your** scene in the Device Central emulator, select **Control > Test Movie**. Press the Enter key on the emulator to see the game move from the title scene to the blank instructions scene. Click on Flash to start work on the next scene.





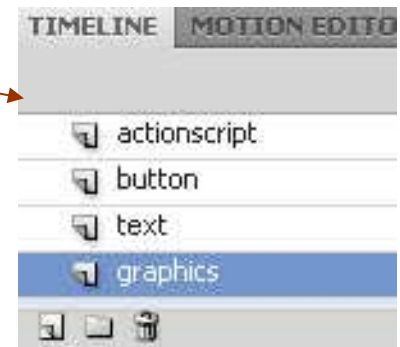
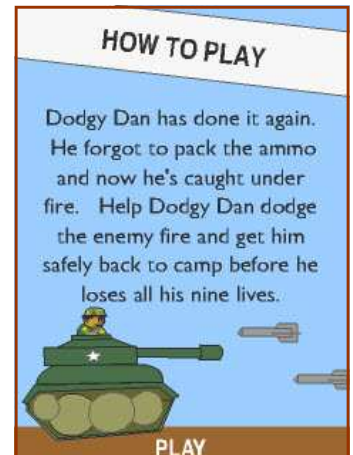
CREATING THE INSTRUCTIONS SCENE

The instructions scene sets up the context for the game and may also contain text that explains how the user can play the game. You should have already planned the instructions you will add for your game. If not, take a few moments to do so. Once you have them, follow these steps to build this scene.

21. Select the **instructions** scene from the Scene panel (shift + F2) to start editing the instructions scene.
22. Just like in the title scene, create the layers for this scene. To add new layers, click on the **New Layer** button.

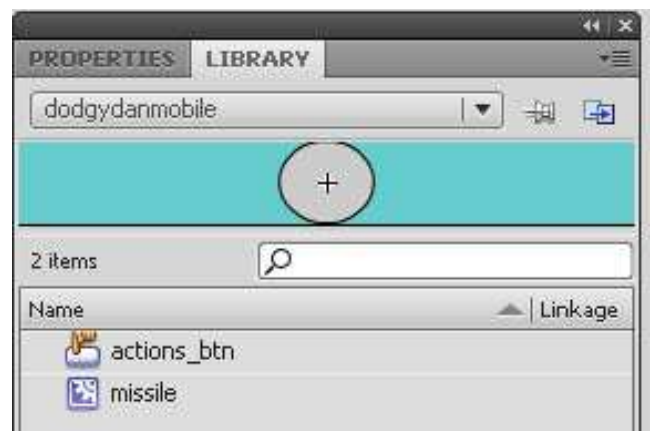
Rename each layer by double clicking on each layer name and typing in the text. Your layers should be named as shown.

23. Select the **graphics** layer and **draw the images** you want in your instructions scene.
24. Select the **text** layer and **add the text** for your instruction scene. This should include the word **Play** in the bottom centre of the design (as shown above).
25. It's now time to **add a button** to the button layer. We will use the same button that we used in our title scene.



Click on the button layer and then select **Window > Library** (or Ctrl + L) to open up the library. You will now be able to see the button you created.

Click on the button and drag it into your scene. Place it to the right of your design just like you did in the last scene. Remember that the player never touches this button, it just holds the script we need to add to allow the player to press the Enter button.



26. Select the **button** and then **press F9** to open the **Actions Panel**.

Enter the following script. This script tells the Flash Lite Player to take the player to the first frame of the game scene.

```
1 on (keyPress"<Enter>") {
2     gotoAndPlay("game",1);
3 }
```





FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

27. It is time to add the final script for our **instructions** scene. This is the script that will stop the player at this scene until they press Enter. It is also the script that will set the initial values for our timer and players lives.

Select the first frame of the **Actionscript Layer**. Press **F9** to open the Actions panel. **Type in the following script.**

```
1 stop();
2 lives=9;
3 timer = 0;
```

You have just completed building your instructions scene. You can test your movie in the emulator by selecting **Control > Test Movie**. Once you have done this, return to Flash to begin work on your game scene.

CREATING THE GAME SCENE

The game scene is the most complex scene that you will create. This is because it is the main scene in your game. It is the scene where you have a character that has to move to avoid being hit by falling objects. It also contains a timer and a counter that will show the number of lives the player has left.

Let's start building this scene.

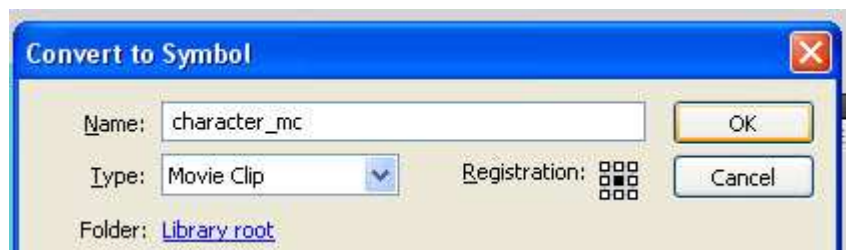
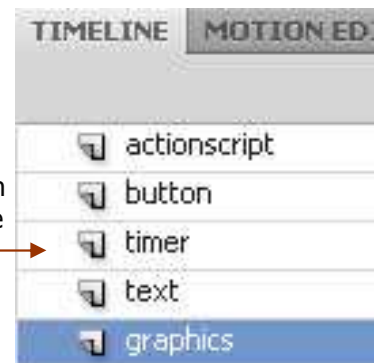
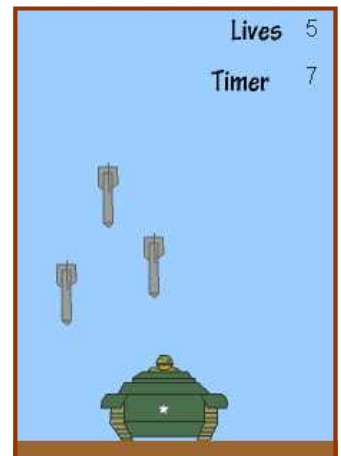
28. Select the **game** scene from the Scene panel (shift + F2) to start editing the game scene.

29. Just like in the title scene, create the layers for this scene. You will need five layers in this scene as we will have a new layer for the timer.

To add new layers, click on the **New Layer** button. **Rename each layer** by double clicking on each layer name and typing in the text. Your layers should be named as shown.

30. Select the **graphics** layer and on this layer draw any **background elements** and the **character** and one of the **falling objects**.

31. Convert the character to a symbol by **selecting it** and pressing **F8**. In the Convert to Symbol box, type in the name **character_mc** and select the **movie clip type** and the **centre square** in the **registration** option.

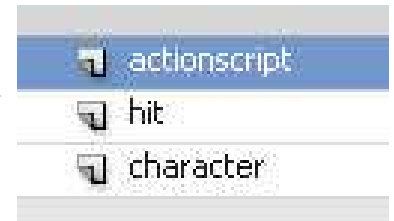




32. We are now going to edit your **character_mc** to prepare it for use in the game.

Double click on the **character_mc** movieclip to edit it. You can tell you are editing the character because your timeline will be different and you will see the following to the top left of the stage. This shows that you are in the game scene and are editing the character_mc symbol.

33. We are now going to set up the layers within this symbol. **Add two new layers** so that you have three altogether. **Rename your layers** as I have here ensuring that you name the one with the character on character.



34. We are not going to make any changes to the first frames in the character or hit layer. We will simply **add some actionscript** to the actionscript layer. Click on the first frame in the actionscript layer and press **f9 to open the Actions panel**. Enter the stop action shown here.



This will make the character appear as it does in the first frame of the character layer until another event in the game (the character being hit) makes Flash move to the next frame.

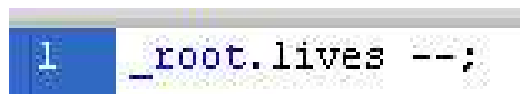
35. Add **new keyframes** to frame 2 of each layer by **clicking on frame 2** of each layer and pressing **F6**.

36. Click on **frame 2 of the hit layer** and in this layer draw the change you want for the character when it is hit. In my game, this is an explosion.



37. Click on **frame 2 of the actionscript layer** and **press F9** to open the **Actions** panel.

Type in the following script. This line will take 1 life off the amount of lives when the character is hit by one of the falling objects.



38. We are now going to add two new blank keyframes in frame 3 of the character and hit layers so that the tank disappears after the explosion occurs.

To do this, select frame 3 on the **character** and **hit** layer and select **Insert >Timeline > Blank Keyframe**. Your timeline should now look like this.





FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

39. We're almost finished setting up the character movie clip. Our final steps involve adding some keyframes further along the timeline and some script which will tell Flash that when it reaches these frames to go to and stop at the first frame where the character is not hit.

To do this, **select all three layers** in the character movie clip and **press F6** to add new keyframes.

Select the keyframe in frame 11 of the actionscript layer and **press F9** to open the Actions panel. **Enter the following script:**

```
1 gotoAndStop(1);
```

40. The timeline in your **character_mc** symbol should look like this.

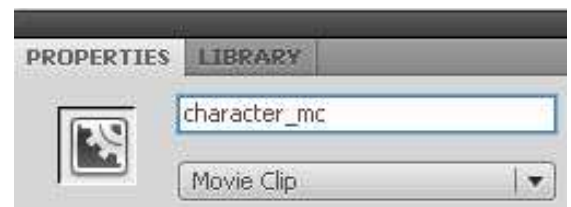


We have now finished editing our character_mc symbol. Click on the **game link** at the top left of the stage to return to the main stage.



41. It is time to give our character an instance name. This is because in a later step we will add some script that refers to the character and it needs to have an instance name for the script to 'find' it.

Select your character and then in the **instance name of the properties panel**, type in the name **character_mc**.

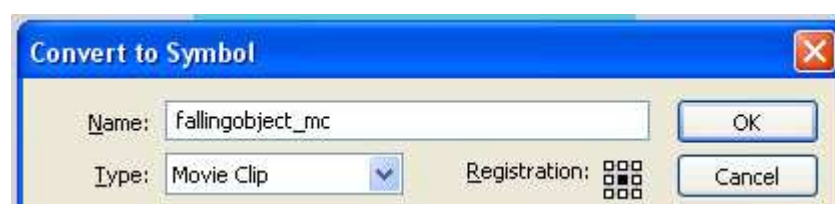


42. Place your **character_mc** at the bottom centre of the stage.



We're now going to prepare the objects that will fall in the game.

43. **Select the falling object** you have drawn on the graphics layer and press **F8** to convert it to a symbol. Name it **fallingobject_mc** and select the **Movie Clip** type and the centre **registration** square. Then click **OK**.



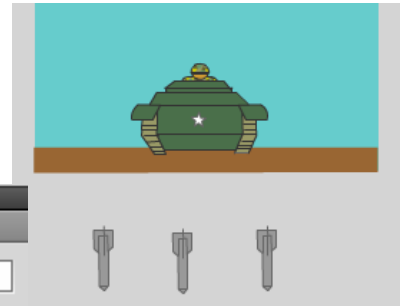
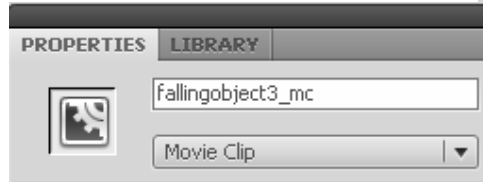


FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

44. Make **two copies** of the fallingobject_mc by copying and pasting the symbol. Place all three copies underneath the bottom of your game.
45. We now need to give each copy its own **instance name**. Click on each copy and in the instance cell in the Properties panel, give each an instance name.

The names should be:

fallingobject1_mc
fallingobject2_mc
fallingobject3_mc



46. It's now time to add the text for the game. Click on the **text layer**, and **add two text boxes**. In the first, type in the word **Lives**. In the second, type in the word **Timer** (as shown here).
47. It's now time to add two dynamic text boxes that will hold the number of lives and the amount of time.



Add the lives dynamic text box first by **adding a text box** next to the word Lives and **typing in the value 0**. (The value isn't important as you will remember we added some actionscript in an earlier step to set the value)

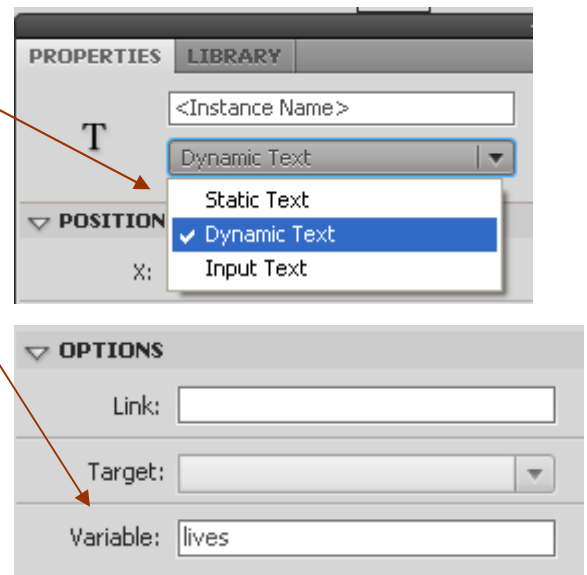
Select this textbox and in the Properties panel make the following changes:

- a) Change the text type to **Dynamic Text**.
- b) In the **Options section** of the Properties panel, enter the word **lives** in the Variable cell.

Now add the timer dynamic text box first by **adding a text box** next to the word Timer and **typing in the value 0**.

Select this textbox and in the Properties panel make the following changes.

- a) Change the text type to **Dynamic Text**.
- b) In the **Options section** of the Properties panel, enter the word **timer** in the Variable cell.



If you test your movie now by selecting **Control > Test Movie** you will see that the values in these text box dynamically change to 9 and 0 when the game scene is opened.



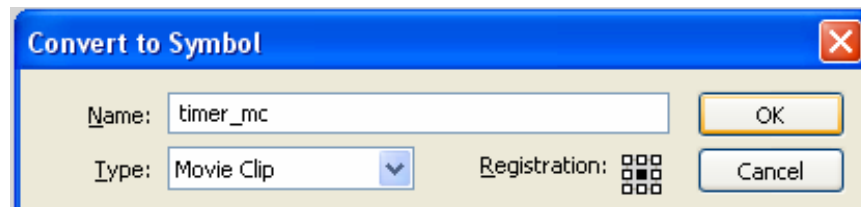


FLASH CLASSROOM TUTORIAL—MAKE A DODGE-EM GAME FOR YOUR MOBILE IN FLASH CS4

48. It's now time to make a timer movieclip. This movieclip will have 13 frames in it. In frame 13, there will be some actionscript to tell Flash to add 1 second to the value of the Timer variable we just set up.

To make your timer movieclip, **select the text tool** and then in the **Properties panel** change the text type back to **Static Text**. Create a **new text box** off the **side of the stage** and type in the word **timer**.

49. Select this new text box and press **F8** to convert the text to a symbol. Name the symbol **timer_mc** and select the **Movie Clip** type. Click **OK**.



50. We are now going to edit this symbol and set it up so that it will continually play and add 1 second to our timer variable each time it hits frame 13.

Double click on the timer_mc movieclip to edit it.

51. In the timeline of the timer_mc, click in **frame 13** on Layer 1 and **F6** to add a new keyframe.

52. **Select the keyframe in frame 13** and press **F9** to open the **Actions Panel**.

Type in the following script:

```
1  _root.timer ++;
```

This script will run each time Flash reaches frame 13 in this movieclip. This will happen every second as the Flash file should be set up to run at 12 Frames Per second (FPS). Ensure your file is set up to run at 12 FPS by clicking on the stage with the black arrow selection tool and checking the value in the Properties panel. If it is different to 12FPS, double click on the text and enter 12FPS.



53. Your timer should now work. Select **Control > Test Movie** to see it in action in the game scene.

54. Click on the **game link** at the top of the stage to return to the main timeline.



55. It's now time to add our actions_btn to this scene and to add the script that will enable the player to make the character move left or right.

To begin, select the button layer and press **Ctrl + L** to open the **Library**. Drag a copy of the **actions_btn** onto the side of the game.



56. Click on the **actions_btn button** and press **F9** to open the **Actions Panel**.





57. Add the following script:

```
1 on (keyPress "<left>") {
2   character_mc._x= character_mc._x-20;
3 }
4
5 on (keyPress "<right>") {
6   character_mc._x= character_mc._x+20;
7 }
8
```

The first part of the script tells Flash to move the character 20 pixels to the left each time the left button on the mobile device is pressed.

The second part of the script tells Flash to move the character 20 pixels to the right each time the left button on the mobile device is pressed.

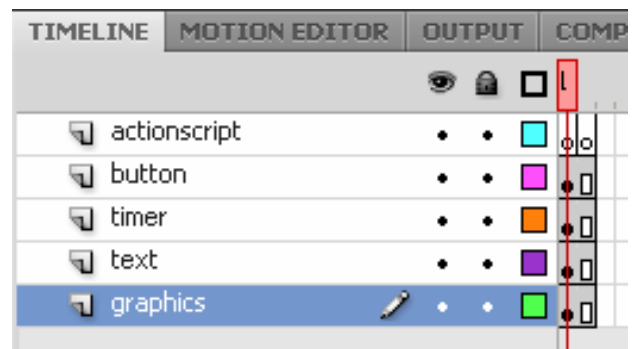
58. Select **Control > Test Movie** to test your game. You should be able to move your character left and right using the left and right buttons.

We are making some significant progress with our game. If you have got everything working so far - well done! If not, go back and double check your script.

It's now time to create the final parts of the game scene.

59. In the second frame of the **graphics, text, timer** and **button** layers, press **F5** to add a second **frame**.

60. In the second frame of the **actionscript** layer, press **F6** to add a **new keyframe**. Your timeline should look like this.



61. It's now time to add the bulk of the actionscript that will make our game work. This script will go in the second frame of the **actionscript** layer.

Select **frame 2** of the **actionscript** layer and **press F9** to open the Actions panel. **Type in the script shown** into the panel.

```
if (lives == 0){
gotoAndStop("gameover",1);
}
```

The first part of the script tells Flash to take the player to the gameover scene if they have lost their 9 lives.

```
if (lives > 0){
gotoAndPlay("game",1);
}
```

The second part of this script tells Flash to go to and play the first frame of the game scene if the value of lives is greater than 0. This means the game will continue.

```
if (timer == 30){
gotoAndStop("youdidit",1);
}
```

The third part tells Flash to go to the youdidit scene if the player lasts thirty seconds without losing all their lives.

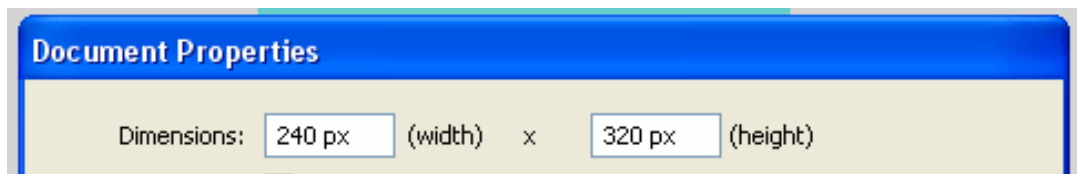




62. It is now time to add the script that will make each instance of the falling object movie clip symbol fall.

Before you add the script, you will need to do a couple of calculations. Firstly, you will need to work out what the height of your game and then add 20 to this to work out the y coordinate at the bottom of the game which will be the point which tells Flash to take the falling object back above the top of the game when the falling object reaches that point.

To work out this coordinate, select **Modify > Document Properties** to open up the panel shown below. If your dimensions are the same, it means the mobile device you have chosen has the same dimensions as mine. If this is the case you won't have to make any changes when you enter the script below. If your dimensions are different, **write them down on a piece of paper** and then underneath these dimensions **take away 20 from the width** to give you a new number (in my case 220) and **add 20 to the height** (in my case 340). Make sure you don't change the dimensions in the document properties box.



Now that you have written down the additional dimensions for the width and height, add the following script to actions panel underneath the script you just entered. If your dimensions are different to mine, make sure you replace the numbers highlighted below with your additional dimensions

```
if (fallingobject1_mc._y>340){  
    fallingobject1_mc._visible = 1;  
    fallingobject1_mc._x= random(220);  
    fallingobject1_mc._y= -20;  
}
```

Change this number to the height of your game + 20.

Change this number to the width of your game - 20.

```
if (fallingobject1_mc._y >= -20){  
    fallingobject1_mc._y += random(10)+10;  
}
```

The script you have just entered works by firstly checking if the falling object is over 20 pixels below the stage. If it is, it makes the falling object visible. We need to do this as we will add some script soon that makes the falling object invisible if it hits the character. The script also changes the x position of the falling object to a random coordinate between 0 and the number you entered (e.g. 220). The final line in the first section of the script tells Flash to then move the falling object to the y coordinate of -20 which is just above the stage. As a whole the first section of the script above makes Flash detect if the falling object has fell lower than the bottom of the game and if so, it repositions that object in a random location above the stage and makes it visible so it is ready to fall again.

The second part of the script checks if the falling object has been returned to above the top of the game and if so, it makes it fall at a random amount of y pixels between 10 and 20. This makes the falling objects appear and fall at different times and speeds.





63. Test your game by selecting **Control > Test Movie**. Providing you have entered your script correctly and also given your falling objects the correct instance names, you should have one falling object that falls down the game and then goes back up to the top and falls again.
64. This is a bit boring though because it doesn't take much effort to avoid the falling object. To make the game more challenging, **copy and paste the script you just entered twice** and then **replace fallingobject1_mc with fallingobject2_mc** and **fallingobject3_mc**. Your script should now look like that shown on the right.

65. **Test your game again**. You should see three falling objects falling down your game scene.

66. We've now got a game where we have our falling objects falling and our character being able to be moved left or right by the player. The problem is at present, nothing happens if the character is hit by one of the falling objects.

In this step, we will add the script that will test to see if the character has been hit by the falling objects and if so, it will go to and play the second frame within the character symbol (e.g. the frame where you put the explosion or something that shows what the character looks like when hit). This script will also make the falling object invisible so it looks like it has disintegrated or been collected.

Underneath the script you just entered into frame 2 of the Actionscript layer, enter the following script.

```
if(character_mc.hitTest(fallingobject1_mc))
{
    character_mc.gotoAndPlay(2);
    fallingobject1_mc._visible = 0;
}

if(character_mc.hitTest(fallingobject2_mc))
{
    character_mc.gotoAndPlay(2);
    fallingobject2_mc._visible = 0;
}

if(character_mc.hitTest(fallingobject3_mc))
{
    character_mc.gotoAndPlay(2);
    fallingobject3_mc._visible = 0;
}
```

```
if (fallingobject1_mc._y>340){
fallingobject1_mc._visible = 1;
fallingobject1_mc._x= random(220);
fallingobject1_mc._y= -20;
}

if (fallingobject1_mc._y >= -20){
fallingobject1_mc._y += random(10)+10;
}

if (fallingobject2_mc._y>340){
fallingobject2_mc._visible = 1;
fallingobject2_mc._x= random(220);
fallingobject2_mc._y= -20;
}

if (fallingobject2_mc._y >= -20){
fallingobject2_mc._y += random(10)+10;
}

if (fallingobject3_mc._y>340){
fallingobject3_mc._visible = 1;
fallingobject3_mc._x= random(220);
fallingobject3_mc._y= -20;
}

if (fallingobject3_mc._y >= -20){
fallingobject3_mc._y += random(10)+10;
}
```





67. If you **test your game** now, you should see that it the game scene is basically complete.

At present, you can move your character left and right to avoid the falling objects. You can also see all three falling objects falling and the character change to the hit appearance when hit. You will also see the timer increasing each second and the lives decreasing each time the character is hit.

There is **one problem** with the game though. Can you work out what it is? It is a way that the player can cheat.

Basically at present, the player can cheat by moving the character to the far left or right of the stage. There is currently nothing stopping the player from just sitting the character off the edge of the game until the timer reaches the set 30 seconds.

We are now going to change this by entering the final piece of script for this scene. This script detects if the character is positioned beyond the edges of the stage and if so moves the player back into the game area.

Enter the following script into the **Actions Panel** below the other script you have entered. Note that if your device has a different width to mine, you will need to change the values of the numbers highlighted.

```
if (character_mc._x < -25) {  
    character_mc._x = 15;  
}
```

This number should be the width of your game + 10.

```
if (character_mc._x > 250) {  
    character_mc._x = 220;  
}
```

This number should be the width of your game - 20;

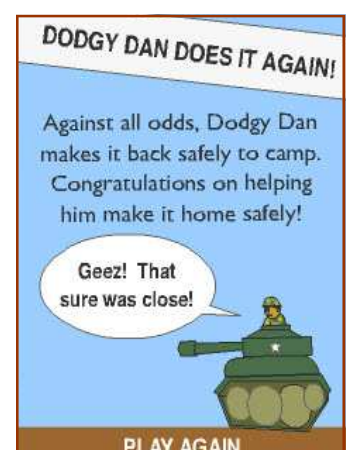
68. **Test your game** again by selecting **Control > Test Movie**. Providing your script is correct, you should be able to see that your character can not be moved off the edge of the game.

Congratulations. You have just finished working on your game scene. It's now time to move on and work on the you did it and game over scenes.

CREATING THE YOU DID IT SCENE

This scene contains the artwork and message for the player that tells them they have successfully made it through the game. It also contains the text 'play again'.

69. To edit this scene, click on the **youdidit** scene in the Scenes Panel.
70. We will now set up the **layers** for you to create the graphics, text, button and actionscript for this scene on.





Add **two new layers** and rename each layer by double clicking on each layer name and typing in the text. Your layers should be named as shown.

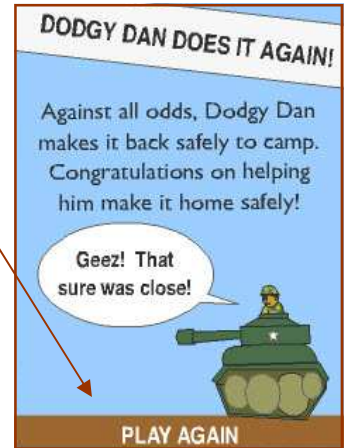
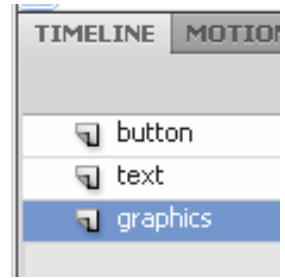
71. Select the **graphics** layer and **draw the images** you want in your **youdidit** scene.
72. Select the **text** layer and **add the text** for your youdidit scene. This should include the words **Play again** in the bottom centre of the design (as shown here).
73. It's now time to **add a button** to the button layer. We will use the same button that we used in our other scenes.

Click on the button layer and then select **Window > Library** (or Ctrl + L) to open up the library. **Click on the actions_btn and drag it into your scene.** Place it to the right of your design just like you did in the other scenes.

74. Select the **actions_btn** and press **F9** to open the **Actions Panel**. Enter the following script:

```
on (keyPress"<Enter>"){
    gotoAndStop("title",1);
}
```

This script makes it possible for the player to return to the start of the game when they press the Enter key on their device.



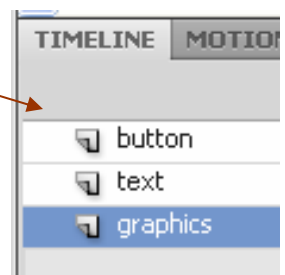
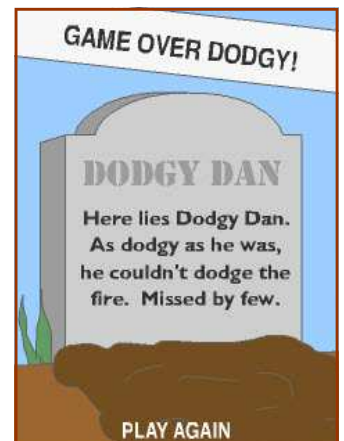
CREATING THE GAME OVER SCENE

This is the scene the player is taken to if they are hit by too many of the falling objects and lose all of their lives prior to thirty seconds being reached.

75. To edit this scene, click on the **gameover** scene in the Scenes Panel.
76. We will now set up the **layers** for you to create the graphics, text, button and actionscript for this scene on.

Add **two new layers** and rename each layer by double clicking on each layer name and typing in the text. Your layers should be named as shown.

77. Select the **graphics** layer and **draw the images** you want in your **gameover** scene.
72. Select the **text** layer and **add the text** for your gameover scene. This should include the words **Play again** in the bottom centre of the design.





73. Select the **button layer** and add a copy of the **actions_btn** to the side of the stage as you have done in previous scenes.
74. Click on the **actions_btn** and press **F9** to open the Actions Panel. Type in the following script:

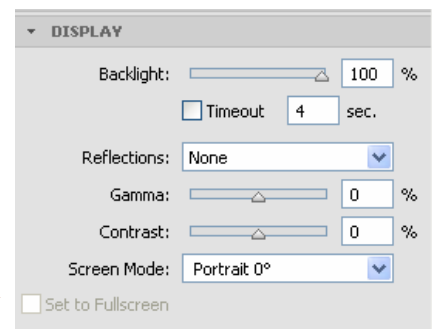
```
on (keyPress"<Enter>"){
    gotoAndStop("title",1);
}
```

TEST YOUR GAME

Your game should now be finished and ready to play. You will need to test your game to ensure that the player is taken to the correct scenes when they make it to thirty seconds and when they lose all of their lives. You may also like to get others to play your game and provide feedback. You may find that your game is too easy or too hard to play. If this is the case, you can change the values of the amount of time from thirty seconds to a smaller or larger amount of time. You could also make the number of lives smaller or larger.

TEST YOUR GAME

Depending on the device you have chosen, you may have the option when you test your game to set your game to fullscreen when it plays on a mobile device. This option is in the Display part of the panel on the right in Device Central. If you can, tick this option. This will make Flash resize to fill the whole screen of devices that your game has not targeted and improves the chances of your game being able to be played on other devices. →



SAVE AND PUBLISH YOUR GAME

75. Whilst you should have been saving your work regularly whilst working through the tutorial, you should now save the final game again. To do this select **File > Save**.
76. You can now also publish your game. To do this select **File > Publish Settings** and choose the file formats you want. The swf file is the format you need for your mobile device. Tick the file formats you want and then select **Publish** and **OK**.

SHARING YOUR GAME

Now that you've created your first mobile game, you'll no doubt want to run it on your phone and share it with friends. If your laptop or computer has bluetooth, you should be able to simply bluetooth the .swf file to your device. On a PC with bluetooth, you can do this by simply **selecting the file and right clicking** and selecting **Send to > Bluetooth device**. You will then be able to find your mobile device and send the file. On a Mac, you can bluetooth files to your phone by selecting **Command + Shift + B**. This will open a window where you can select the device you want to send your file to and then click send to transfer it to your phone.

Once you have it on your phone, you can share it with friends with the Flash Lite player on their phone by selecting the file and then selecting **send via bluetooth**. Your phone will then search for phones within range and you can select your friend's phone and click send.





YOUR CHALLENGE

So you've created your first mobile game and you can now play it on a real mobile. That's pretty cool. But despite all your hard work, you've really just created a game by following the steps I've provided you in this tutorial. Even though that's great, it would be great to see you prove that you can now create your own game by yourself.

Below are a list of challenges that I have set for those of you who are willing to take on your next challenge with mobile game design. The knowledge and skills you have gained in this tutorial should have prepared you to be able to take on these challenges.

We'd love to see the mobile games you create with the help of this tutorial and in response to these challenges. Share your games with the Flash Classroom community by sending them through to kristine@eq.edu.au.

Challenge #1

Make a copy of your game and change the value of your variables to make the game harder or easier to play.

Challenge #2

Make a copy of your game and in this new copy, completely change the graphics and text in your game to create a new game context. For example, I changed the graphics and some of the variables in my game to convert it into a game called 'Apple Catch' where the player has to catch 20 apples in 30 seconds in order to keep their job. This game is online in the Flash Classroom gallery at www.flashclassroom.com.



Challenge #3

Take the BAR Challenge. This is where you change your game by making something **Bigger**, by **Adding** something and by **Replacing** something.

Challenge #4

Add some extra scenes to your game to create some new levels. Change the variables in each scene to make the higher levels harder to play.

Challenge #5

Add some sound effects or music to your game. To do this, import your sound files (mp3 or wav) to the library via **File > Import to Library** and then drag the sound from the library into the keyframe where you want the sound to start.

Challenge #6

Work in a team with your classmates to create your own website where your friends and other students can download your games. Providing you use your own graphics and media, you could even sell your games at a school fete or as part of a fundraiser. Professional mobile developers sell their games through sites such as the Nokia shop at ovi.com.

