Assignment #6
Battleship (Web application)
MSIS 558, Prof. Robin Burke
Assigned: 4/23/01
Due: 5/7/01

Objective

Create a version of the Battleship game that runs on the web. The project will be done in two parts: one part will be an ActiveX DLL that implements the game by interacting with the database, and the other an IIS application that uses the DLL.

The game DLL will have the following API:

Public Function NewGame () As Integer
Creates a new game in the database and returns its game id.

Public Function NewUser (ByVal Username As String) As Integer
Creates a new user with the given username and return a user id.

Public Function AllGames (ByVal Username As String) As Collection
Returns a collection of all the games in the database. Each entry in this collection should be an instance of the btlGame user-defined type. If the collection is empty, the user is a new user. See below.

Public Function Game (ByVal GameId As Integer) As btlGame
Returns the information associated with the game.

Public Sub Board (ByVal GameId As Integer, ByVal UserBrd As Boolean, _
ByRef Board () As btlCellStatus)
Modifies the Board array parameter to reflect the status of the board.

Public Function Hit (ByVal GameId, ByVal X As Integer, ByVal Y As Integer) As Boolean
Makes the move as indicated by the X,Y coordinates. A true result means the game is over.

Public Function ComputerMove (ByVal GameId) As Boolean
Lets the computer move.

The game DLL will also need to define some public data types: the btlGame user-defined type and the btlCellStatus enum. See below. The databases that back the game component will be the same as in Assignment #5.

The web application will consist of a WebClass with several associated pages. Flow through the application will proceed as follows:

?? When a user connects, she first sees the login page and enters her username.

?? If the user is new, a new game is automatically started.

?? If not, a list of previous games played is shown. The user can click on the link associated with each game id in the table to go to that game, or can start a new game.

?? When a game begins or is resumed, the board page is shown. Each cell on the user’s board is an image. Unseen cells are links that invoke the Hit function described above, and cause the board page to be redisplayed with new data.

?? When the game is over, the board ceases to be clickable.
Activity

The game DLL should be very similar to the game that you implemented in Assignment #5. The WebClass should be very simple, containing only the code needed to process the template tags and create the correct links.

Submission

Students should

Turn in **at the start of class** a hardcopy of the code of the DLL and the application with a cover page clearly indicating the number and name of the assignment and the student’s name and ID #.

Before class time, submit a folder containing two sub-folders – one with the ActiveX DLL project, and one with the web application project. The folder should be named with your last name and the assignment number. For example, student Smith would submit a folder titled “Smith_A6” containing two folders: DLL and Game, each containing a VB project and associated files.

Assessment

This assignment will be assessed on the completeness of the solution to the problem and on the clarity of your program code. Use readable variable and method names. Break large functions and subroutines into small, cohesive ones. Document your SQL and HTML.

Hints and Notes

1. You can adapt my templates or create new ones to customize your game. The templates are available on the “Assignments” page on the course web site. The single-pixel gif images that you will need to use the board template are also available there.

2. For this assignment, assume a fixed board size of 10x10.

3. The entry in each cell in the user’s table will look like this

   `<a href="Battleship.ASP?WCI=Board&WCE=Hit&X=2&Y=1&GameId=1025">`<br>
   `<img src="blue1.gif" width=20 height=20>`

   The first parts of the query string (the part of the URL following the `?`) are created using the URLFor function. You will need to add to it the X and Y locations of the cell and game id. This information can then be recovered inside of your IIS application by accessing the Request.QueryString function as follows:

   ```
   X = Request.QueryString("X")
   ```

   The images are GIF images containing a single pixel of one color. These images are can be rescaled as shown in the example to create a patch of color.

4. I have defined two of the templates that you’ll need: one for the games page and one for the board.

   **a6-games-template.htm**

   WC@Username: Replaced by the username.

   WC@GamesData: Replaced by table rows that enumerate all of the games that a user has in the database.

   WC@NewGameLink: Replaced by a link that will generate a new game.

   **a6-board-template.htm**

   WC@GameId: (One per page) Replaced by the game id from the database.

   WC@GameDate: (One per page) Replaced by the game date.

   WC@Username: (One per page) Replaced by the username.
WC@GameStatus: (One per page) Replaced by a string indicating the status of the game: In Play or Over.

WC@CellImage: (One per computer board cell) Replaced by an IMG tag containing a scaled single pixel gif of the correct color to render the board cell.

WC@CellLink: (One per user board cell) Replaced by either an image tag as in WC@CellImage, or an anchor tag (hyperlink) for the user’s click action, enclosing an image tag.

WC@NewGameLink: (One per page) Replaced by a link that will generate a new game.

5. To develop this project, you can work with two VB projects at once, but if you need to update the DLL, you’ll have to exit the web application project and restart it.

6. The btlGame and btlCellStatus data types should be defined as follows:

   ```vbnet
   Public Type btlGame
       Date As String
       Id As Integer
       Status As Boolean
   End Type
   
   Public Enum btlCellStatus
       btlUnknown = 0
       btlEmpty = 1
       btlShip = 2
       btlHit = 3
       btlSunk = 4
   End Enum
   ```

   Note that the btlShip value should only be returned by the Board function when the board is the computer board, and the user is allowed to see where his ships are. While debugging, you may wish to allow it to be returned for both boards.

7. The only state maintained by this application should be the GameId stored in the board. You will not need to use session objects, cookies, URLData, persistent WebClasses, etc. Your DLL is basically just a wrapper around the database.

8. A Collection is a Visual Basic object that is somewhat like a dynamic array. The AllGames function in the DLL returns a Collection because arrays cannot be used as return values.

9. You must have IIS or PWS (Personal Web Server or under Win2000 Peer Web Services) installed on your machine in order to develop web applications. If you do not, you can still develop the DLL part of the assignment, interacting with it using a regular VB form, until you can work on the WebClass component. Some of the machines in LH 317 are properly configured for web applications so that you can work on your WebClasses during lab time.