Submission for Unity Cloud Intern Assignment

Name: Siqing Xu

Github repo link: https://github.com/SQingXu/UnityCloudServiceAssignment

Frameworks and Library used:

Servlet in Java for running the application on Tomcat Server (Local)

JSoup in Java for parsing the source page content

(Some instructions for running the application: I include the JSoup library jar in my /Webcontent/WEB-INF/lib/, to build application, the library might be needed to include inside the build path. It takes me a long time to find the right way to incorporate the JSoup library into the Servlet.

this blog might help:

<https://stackoverflow.com/questions/11652431/eclipse-add-jar-to-dynamic-web-project/15173716#15173716> )

Overview of my design:

Two source files are in my design (Important functions are listed here):

* ParsePage

1. updateDocument: replace/update the instance variable doc which store the source page content with the content of given url directed.

Used for initialize the data of the class which will later be used repeatedly.

1. getUlElement: get a random element of a given unordered list in the source page. How random number generate are via internal functions

(explained in how the requirement is fulfilled)

* ParsePageServlet

1. Init: when the servlet object is first initialized, a instance of class ParsePage is initialized and retrieve data from the given link Unity Game Gallery
2. doGet: set the current session lifetime to 5 second, if the session expired, the ParsePage will update its source page data from the Website in case the data of the source page is updated
3. service: the function is triggered every time the refresh button is pressed. It will retrieve one element (one game content) from the source page using the ParsePage instance’s getUlElement and directly render the content on the page from by passing in the name of the class which immediately contain the unordered list “div.game-list-wrap.clear”

So the step will be, first a servlet object is created, then the init function create a instance of ParsePage as its own instance variable and retrieve the content from the website. Then, whenever the link is queried or the page is refreshed, the full content of a random game will be rendered on the page. I believe such approach will separate the user query functions from the internal functions in which how a page is parsed. Therefore, the structure is more clear and changes can made more easily.

Fulfilled requirements:

1&2 A different game must be rendered every time the page is refreshed & No game should be repeated until all games have been viewed by that user.

The randomIndex function in ParsePage class meet these requirements. With a given range, which is retrieved from the size attribute of given unordered list of type Elements, a list of integers from 0 to range-1 will be created and shuffled. (see in the function of initRandList) A pointer will loop through the list one by one for every call to the function and reshuffled the list whenever the pointer reached to the end. This prevents repeated game until the user sees all games.

Therefore, the only possibility for consecutive repeated index is when one loop ends and a new loop starts, which can not be the end current list. This is handled by storing the previous accessed index. If previous one and current one equals, remove the current element from the list and append to the end, and retrieve the new element from the same index. This make sure a different game will be rendered every time.

3 The application does not contain any content specific to any one game because the application only use JSoup to parse out the game content on the game list with random index. Source are all based on the doc variable retrieved from the given url.

4 I believe I minimize the user load time and bandwidth usage by storing the source content of the page inside the variable. So, whenever the page is refreshed, instead of retrieve the content from the website again, the game content is parsed from the local copy stored inside the variable. However, the case of page information would be updated is also considered so the local copy is refreshed whenever the session expired.

Potential place of improve:

* The visual layout can be more clear to show the information about the game.

I can use some pre-programmed format and retrieved information to populate the the format.

* I observed that there is a “load more” button at the end of the page. I should find some way to retrieve more games after the button is clicked.
* Currently the way how I find unordered list is by hard-code the class name which I find. More automatic way to detect the location of the list may be implemented
* Optimize the user load time and bandwidth usage by implementing some listener in the page so that the source page content will only be updated if some change to the source page is detected.
* Implement category selection function so that the if user select 3D game option, the page will only randomly show games from category of 3D game.