PLTW COMPUTER SCIENCE

Problem 1.2.6

App Design

Introduction

Sometimes developers are meeting the needs of a client. At other times, developers use their skills for themselves. What do you want to create?

http://www.jacobsschool.ucsd.edu/news/news_releases/id=1340

A UCSD development team that created an app to shorten wait times for people commuting across a border.

Materials

- Computer with Internet access and Google Chrome™
- Project notebook

Resources

Problem 1.2.6 Rubric

Procedure

1. Form pairs as directed by your teacher. Meet or greet each other to practice professional skills. Set team norms.

2. Review the criteria for the project:

   User interaction

   The app should have a clear purpose that will be useful to the user.

   Originality

   The app could be a modification of an existing program, but it should attempt an original improvement or feature. You can find some ideas for starting points for App Inventor projects at the following links. Note that App Inventor Classic and App Inventor 2 projects are in different languages, but the ideas are transferable.

   http://appinventor.mit.edu/explore/ai2/tutorials.html

   http://appinventor.mit.edu/explore/tutorials.html

   http://appinventor.mit.edu/explore/teach.html

   http://www.appinventor.org/tryit

   http://gallery.appinventor.mit.edu/#

3. Review the rubric for this project.

4. Brainstorm ideas using tag lines and thumbnail sketches. Follow the guidelines for brainstorming: go for quantity; never criticize ideas during brainstorming; “piling on” is welcome.

5. Develop one or two of your ideas with further discussion and documentation.

6. Decide on one simple app that you will develop into a product. Using diagrams, lists, and flowcharts, explain to another team of developers what you want the product to do. Up to now, you have been (mostly) playing the role of the client. Now you will transition to the role of developers. When another team tells you what they want their product to do, pretend they are your client. You will be the ones that have to create their product. If they are not giving you enough information to create their product to their satisfaction, ask questions.

7. You will now serve as the software developers for your own project. Plan the first sprint.

   Your plan should include two bullet point lists, as follows:

   - a **product backlog** with the most important **user stories** at the top. User stories lower on the list can be large and poorly defined.
   - a **sprint task list**. Sprint backlog items should be specific and broken into small tasks.

8. Strategize, code, and test in small increments.

   - Switch driver and navigator roles every 10 minutes or so.

- Include comments as you develop your solution.

9. Compiling your program into an APK file uses an additional App Inventor program on the App Inventor build server to create a new program in a lower-level language than your AIA program, so that it can run directly on the Android operating system instead of needing the AI Companion as an **interpreter**. Compilers translate an entire program to a lower-level language *before* it is executed, while an interpreter translates a program line-by-line to a lower level language, while the program is being executed. Lower-level means closer to the electronics. Select **Build** from the menu at the top of the App Inventor environment to build your program into an APK. Install the APK file of your program on an Android device.

10. Write a reflection on how well your product meets the clients' needs. Include descriptions on strengths and weaknesses of the solution as well as potential plans for the next sprint.

11. Prepare to present your project at the end of the first sprint. Your teacher will describe the presentation format they would like you to use.

12. Practice Opportunity for the *Create* Performance Task

    You can capture a video of your program by capture video of the AIA file running on the emulator screen, by using a videocamera with another computing device to record the Android device screen, or by using a rooted Android device mirrored on another machine with video capture software on the other machine.

    “Create a 50-59 second video in which you demonstrate the running of at least one significant feature of your program.” (Adapted from College Board Create Performance Task Part 1.)

13. Practice Opportunity for the *Create* Performance Task

    “Identify the purpose of your program and explain what the video illustrates. (Approximately 150 words)” (Adapted from Create Performance Task Part 2a.)

14. Practice Opportunity for the *Create* Performance Task

    “Describe the incremental and iterative development process you used, focusing on two distinct points in that process. Describe the difficulties and/or opportunities you encountered and how they were resolved or incorporated.” (Adapted from Create Performance Task Part 2b.)

15. Practice Opportunity for the *Create* Performance Task

    Your entire App Inventor program is an algorithm to repeatedly look for events and execute an event handler for each user or program event that occurs and has a handler. Your algorithm for the event loop combines the algorithms of each event handler that you created. “Describe how each algorithm within your algorithm functions independently, as well as in combination with others, to form a new algorithm that helps to achieve the intended purpose of the program. (Approximately 200 words)” (adapted from College Board *Create* Performance Task Part 2c.)

Note: This last four deliverables are adapted from the official College Board Create Performance Task but they do not duplicate the content of College Board Task or Rubric. The task provided here contains elements that are different than the College Board Performance Task and Rubric. Please

reference official College Board materials.

Conclusion

1. Reflect on the creative process you used. What was useful? Discuss your reflection with your partner and then write a reflection individually.
2. Reflect on the team dynamic. What helped the team work well together? Discuss your reflection with your partner and then write a reflection individually.

Problem 1.2.6 Designing an App Rubric

	4	3	2	1
Solves Problem	Artifact fully addresses personal, practical, or societal intent posed by problem statement	Artifact addresses the personal, practical, or societal intent posed by problem statement	Artifact mostly addresses the personal, practical, or societal intent posed by problem statement	Artifact does not adequately address the personal, practical, or societal intent posed by problem statement
Documentation	Uses appropriate documentation of work.The three formats for documenting work: • App Inventor comments • Project design notebook • Named versions of project	Uses appropriate techniques in 2 forms for documenting work	Often uses appropriate techniques for documenting work	Does not usually use appropriate techniques for documenting work
Collaboration	Provides helpful original input to others Promotes positive, productive, and respectful team dynamic Encourages and incorporates input from others Promotes equitable workload	Provides adequate original input to others Maintains positive, productive, and respectful team dynamic Positively incorporates input from others Maintains equitable workload	Significant but limited input Usually maintains positive, productive, and respectful team dynamic Receives input from others Shares workload somewhat equitably	Limited input Is not promoting positive, respectful, or productive team dynamic Discourages or is unresponsive to input from others Does not promote equitable workload
Appropriate Algorithm	Code demonstrates use of appropriate algorithms	Code mostly uses appropriate algorithms	Code often uses appropriate algorithms	Code does not use appropriate algorithms
Presentation	Effective presentation techniques: Posture Gestures Voice	Mostly effective presentation techniques: Posture Gestures Voice	Adequate presentation techniques: Posture Gestures Voice	Inadequate presentation techniques: Posture Gestures Voice

	Eye Contact	Eye Contact	Eye Contact	Eye Contact
Explanation of Algorithm	Comments clearly and thoroughly explain the algorithm(s)	Documentation explains the algorithm(s)	Documentation insufficiently explains algorithm	No documentation
Explanation of Problem Solution	Prose clearly and thoroughly explains how the solution meets the need Prose clearly explains the solution's strengths and weaknesses and strategizes for improvement	Prose explains how the solution meets the need Prose mentions a strength or weakness and ideas for improvement	Prose explains how the solution meets the need Prose mentions a strength or weakness	Prose does not address how the solution is connected to the need
Planning	Product backlog shows intent Sprint task list subdivides to simpler subproblems	Product backlog shows intent Sprint task list shows subdivision to simpler subproblems, but some tasks are too big	Product backlog and sprint task list show inadequate attempt to indicate long-term intent and an attempt to break down problem into simpler tasks	Product backlog or sprint task list are missing

Other comments: