|  |  |
| --- | --- |
|  | Concept Assignment 5  PLTW Computer Science CSP Core Training |

# Event-Driven Programming

|  |  |
| --- | --- |
|  | Learning Objectives |

LO5.1 While working through Activity 1.5.1, the teacher will:

* Learn about the principles of Human Computer Interaction (HCI).
* Evaluate an interface based on HCI principles.
* Practice completing a portion of the CS Principles Explore Task.

LO5.2 While working through activities in Lesson 1.5, the teacher will:

* Become familiar with using Tkinter in *Python*®.
* Modify basic Tkinter programs to change the behavior.
* Create new features in Tkinter program.

LO5.3 While completing Problem 1.5.4, the teacher will:

* Experience using the Agile Scrum method to create and develop a project.
* Explore their creativity in designing a user interface (UI) in Tkinter.
* Share the product they developed with the group.

|  |  |
| --- | --- |
|  | Introduction |

### Overview of Event-Driven Programming

Students enjoy creating programs that respond to the user, which makes event-driven programming useful for the classroom. Programs written in Scratch and MIT App Inventor tend to be event driven, though they don’t have to be. An event-driven program contains distinct parts:

* One initialization procedure that’s executed when the program is started.
* Event handlers which can be triggered by a user event like a keypress or mouse motion.

Event-driven programming is one of several programming paradigms used in the course:

* Event-driven programming emphasizes event handlers that are triggered by events.
* Procedural programming emphasizes the sequential execution of instructions, organized into procedures that can be called. When creating a Lightbot program or analyzing a single stack of Scratch or MIT App Inventor blocks, you are thinking within this paradigm.
* Functional programming emphasizes functions for transforming data. Work in the iPython session in Lesson 1.3 has this feel, because the programmer sees the results of calling a function before deciding what function to create or call next.
* Object-oriented programming emphasizes the use of objects with properties and methods.

### Overview of Event-Driven Programming in the Course

* Lesson 1.1 develops event-driven programming concepts in Scratch. Event handlers are introduced in Activity 1.1.3.
* Lesson 1.2 activities use event-driven programming in App Inventor.

Both Scratch and App Inventor are primarily used within an event-driven programming paradigm. Event-driven programming languages—like Scratch and MIT App Inventor—are known to obscure to beginners the initialization code, much of which is not represented in the stacks of blocks. Emphasizing the initialization of an event-driven program provides clarity. In App Inventor, for example, emphasize that the Designer view represents code as well, and that the components are created and initialized using the Designer view and properties. Only after this initialization does the AI Companion begin looking for events to trigger event handlers.

* Lesson 1.5 develops concepts of event-driven programming using the *Python* Tkinter library.
* Activity 2.2.1 offers students the opportunity to explore JavaScript® for a web client. In the context of programming a web page, JavaScript is usually used in an event-driven paradigm.

|  |  |
| --- | --- |
|  | AP CSP Enduring Understandings (EU) and Learning Objectives (LO) |

Lesson 1.3 begins with the processor and works up through the high-level language of *Python*. Lesson 1.4 continues into higher levels of abstraction with object-oriented libraries. Lesson 1.5 extends the story line of abstraction from the previous two lessons.

In Lesson 1.5, even higher level abstractions are used: the components of user interfaces (UI). If you include Lessons 1.1 and 1.2 in your instruction for this course, Lesson 1.5 could be left out of the curriculum with the exception of Activity 1.5.1 Human-Computer Interaction. For students who’ve been introduced to code in earlier courses, you can implement the full Lesson 1.5 after you begin this course with Lesson 1.3.

Focus on the following concepts. (The codes refer to the Enduring Understandings and Learning Objectives from the College Board’s CS Principles framework.)

* Multiple levels of abstraction are used to create programs. EU2.2 through LO2.2.2 [P3], EU4.2 through LO4.2.4 [P4].
* Programming is often a collaborative process. EU5.1 through LO5.1.3 [P6].

|  |  |
| --- | --- |
|  | Part 1: Create Python UI |

Your instructor will lead the group through the following activities using the course materials. Specific steps from student activities are referenced to facilitate your work.

1. Work through Activity 1.5.1 and discuss items 1–16 with your partner or in a small group (nothing to submit).
2. Complete step 17, which is practice for the CS Principles Explore task using articles from the ACM TechNews archive or another source.

|  |
| --- |
| Submission Item |
| 1. Activity 1.5.1: Submit either Task Part 1 (artifact) or Task Part 2 (essays) as directed by your Master Teacher. |

1. Work through Activity 1.5.2 with your partner to become familiar with Tkinter in *Python* and the API documentation (nothing to submit).
2. Work through Activity 1.5.3 and modify the provided programs as indicated in steps 11, 17, and 22.

|  |
| --- |
| Submission Item |
| 1. Activity 1.5.3: Submit working code for steps 11, 17, and 22. |

1. Design a *Python* UI with Problem 1.5.4. Document your process with a backlog, a sprint task list, screenshots of your UI, and your *Python* code. Communicate the idea of the program you created in a gallery walk. Note that Problem 1.5.4 has students play the role of client in brainstorming a project and then pass the idea on to a separate development team.

|  |
| --- |
| Submission Item |
| 1. Problem 1.5.4: Submit documentation and working code for UI. |

|  |  |
| --- | --- |
|  | Part 2: Considering Classroom Implications |

1. Discuss strategies for pairing students and for managing pair-programming dynamics.
2. Increasingly in this course, students will be asked to do tasks where they’re not given all the information they need to complete the task. In Lesson 1.5 they need to explore the documentation for Tkinter to determine how to complete many of the exercises. How will you help students learn to use the documentation and other resources available instead of just asking for help when they get stuck?

|  |
| --- |
| Submission Item |
| 1. Write a reflection about the things you’ve learned today. Consider highlighting new things you’ve learned, items you need to consider for implementing this in your classroom, and ideas and suggestions you’ve heard from others. Use the questions in Part 2 of the assignment as prompts, but don’t feel limited or constrained by just those questions. |

**Trademark Attribution** PLTW, Project Lead The Way, and the PLTW logo are registered trademarks of Project Lead The Way, Inc. "Python" and the Python logos are trademarks or registered trademarks of the Python Software Foundation. All other brand names, product names, or trademarks belong to their respective holders.