A Framework for Understanding and Evaluating the Development of Computational Thinking

Prepared by the Scratch project team at the MIT Media Lab

What does it mean to be a *computational thinker*?

It is not simply a matter of knowing how to use computers. Rather, it means being able to use computational ideas and strategies to understand and describe how things behave and interact. Computational thinking is useful for describing actions and interactions in many different types of systems – cars on a highway, animals in an ecosystem, characters in an adventure game.

To develop as computational thinkers, students need to become fluent with a set of *computational concepts* as well as set of *computational practices*.

Computational Concepts

Computational concepts are particularly useful for describing processes – that is, for explaining how things change and evolve over time.

Examples of computational concepts include:

- *sequence*: putting actions together in a specified order
- *parallelism*: executing different actions at the same time
- *events*: actions that trigger other actions
- *conditionals*: selecting between actions based on whether a condition is true
- *variables*: storing data that can be accessed and changed over time
- *synchronization*: coordinating multiple threads of activity

Computational Practices

To put computational concepts into action, students need to learn *design practices* for creating computational artifacts (such as interactive games and simulations) and *social practices* for collaborating with others on the design and use of computational artifacts.

Examples of *design practices* include:

- *problem finding*: identifying personally-relevant issues and challenges
- *experimenting*: trying out different possibilities
- *debugging*: systematically figuring out what went wrong
- *modularizing*: dividing work into meaningful chunks
- *reflecting*: thinking about the implications and applications of what you've done
- *iterating*: making modifications and revisions – and trying again

Examples of *social practices* include:

- *sharing*: making your work available to others
- *remixing*: revising and building on the work of others
- *crediting*: providing appropriate acknowledgement to others
- *critiquing*: provide feedback on the work of others
- *co-creating*: designing and developing in collaboration with others