This is the coding scheme guide for identifying actions, episodes, and subgoals. There are 4 steps in the coding process.

Step 1: Codifying the Transcripts

Look at the Screen Capture videos for each participant and transcribe in the following fields in the given format:
Description - a description of what they are doing, within which file or application. Be as detailed as possible.
Quote - What the participant was saying while performing the actions
Action Code - Will be completed in **Step 2**
Episode and Subgoal - Will be completed in **Step 3**

Example Transcription

Index	Time stamp	Description	Quote	Action Code	Episode	Subgoal
1	hh:mm:ss	looking for a code file xx within package explorer	"I want to open this package explorer here so I can figure out where I want to put this in"			
2	hh:mm:ss	Pasted ‘defn-get-model’ from earlier copy into equinox-analysis.symbolic-executer.set-caller	“I have to get a copy of both the ‘officiality’ and the ‘model’ but this is some non-versatile. I’ll put it here now.”			

Step 2: Codifying the Actions

Update the Action Code in the Transcription Document.

Use the following tables to identify the action for each row in the transcription document.
***Note*: Each (transcription) row might have more than one action associated.

Action Code Definitions

Codes	Definitions
Read	Examining information from artifacts (e.g. code, documentation, terminal output)
Edit	Any change made directly to code or related artifacts.
Navigate	Moving within or among artifacts (e.g. pulling files from Git, opening files, scrolling through a file).
Execute	Compiling and/or running code.
Ideate	Constructing a mental model of future changes.

Action Code Rulebook

Here are some examples of the action codes:

Actions / Intentions	Instances	Positive Examples	Negative example
read	Reading code, documents, etc. Reading error messages or output. Examining VCS output.	Reading error: extra-var-decls Read stack overflow answer	Copies 'defn-collect-all-substring-ast' and 'defn- primitive-type' Move to package explorer to find file
edit	Permanently changing the code. Experimental or temporary changes to code. Staging or committing to the remote repository.	modifies the [x] attribute by adding % classList property value "xx" and deleted it added a console log statement	Search for 'this.class' within file xx.ts Print variable value directly from terminal
navigate	Move to a different window outside the IDE. Move to a different file within the IDE (through package explorer or hotkeys) Move to a different application. Launch a build/environment (ex. Loading REPL)	Clicked on the browser window in the background Closes out Terminal window Search for 'this.class' within file xx.ts	Scroll through the output or code.
execute	Running code to validate models Pushing to VCS repository	Types python run.py, presses enter. Starting nREPL server from file X	added a console log statement
ideate	Constructing mental models of future changes, locating a bug, reason about behavior. Hypothesizing	"To start, I'm going to take a look at what this component [x] is so that I can more easily find it in the IDE." "Alright I am going to check., what I have done"	"I'm using the console to see if the data is showing up correctly"

Step 3: Identifying Subgoals and Episodes

Subgoal: What the participant is trying to achieve. Eg. understand why the code has to be refactored.
This is like a topic sentence but doesn't need to have all the details of the participant's actions.

- Identify what they are trying to achieve using the 'Quotes' and 'Description' column of the transcription.

**Note*: If the participant comes back to a subgoal they previously worked on, please make sure that 'Subgoal' field matches the previous occurrence.

Episode: A series connected actions that are performed continuously (temporally related) to complete a specific subgoal.

- Assign a unique number to each episode. (ascending order)

**Note*: Episode number changes every time the participant changes a subgoal.

Here is an example of the coded transcription:

Index	Time stamp	Description	Quote	Action Code	Episode	Subgoal
97	15:47	Looking for query example, Explaining why example query is needed.	“...a lot of our queries return individual statements, so to get a nice slice I had to email a colleague to ask for a particular query that will return just a program slice ...so I am going to test that now.”	ideating	30	Checks email for example queries for slice representation to translate to Cobol, familiarize with query interface
...						
100	17:04	error pops up on the emacs window. reads the error message.	"I am starting up the repl again for the correct project."	ideating	31	Start Repl for correct Project
...						
104	17:37	Examining query to familiarize with code.	“I haven't interacted with the query interface in a couple months so I am going to familiarize myself with the code.”	read	32	Checks email for example queries for slice representation to translate to Cobol, familiarize with query interface
...						
108	18:32	Error message appears in emacs message pane indicating a subprocess failed.		read	33	Start Repl for correct Project

Step 4: Identifying Patterns

We use the sequence of episodes to analyze for patterns. This is a ground-up approach, so no predefined codeset is provided.