CS 2334: Project 2
Class Abstraction
A Trial has many States

• A Trial knows how to compute statistics over its list of States
Project 2

Now, we want to think about the list of Trials that belong to a single Infant ...
Project 2

Infant

Trial

State State ⋮ State

Trial

State State ⋮ State

Trial

State State ⋮ State
In what way are Infants and Trials the same?
Infants and Trials contain lists of items about which they can compute statistics!
Project 2

In what way are Infants, Trials and States the same?
Infants, Trials and States can have statistics computed about them!
Project Design

• **SingleItemAbstract:**
  • Classes about which statistics can be computed
  • Contain one or more States

• **MultipleItemAbstract:**
  • Classes that compute statistics
  • Contain a list of items (which implies more than one State)
Project Design

• SingleItemAbstract:
  • Classes about which statistics can be computed
  • Contain one or more States

• MultipleItemAbstract:
  • Classes that compute statistics
  • Contain a list of items (which implies more than one State)
SingleItemAbstract

• Requires implementing classes to provide a way to compute statistics over States
  • For an individual State, this is trivial
  • For a Trial: you have already implemented this code
• Note that a GeneralValue is returned
State

- Similar to project 1
- New: “compute” statistics

```
State

-leftWrist:Point3D
-rightWrist:Point3D
-time:double

+State(strg:String)
+getLeftWrist():Point3D
+getRightWrist():Point3D
+getTime():double
+toString():String
+getMaxLeftWrist(dim:int):GeneralValue
+getMinLeftWrist(dim:int):GeneralValue
+getAverageLeftWrist(dim:int):GeneralValue
```
MultipleItemAbstract

- Requires from implementing classes:
  - The number of component items
  - Access to the individual items
- Provides:
  - Statistics computation over all containing items

```
MultipleItemAbstract

+getSize():int
+getItem(index:int):SingletonItemAbstract
+getMaxLeftWrist(dim:int):GeneralValue
+getMinLeftWrist(dim:int):GeneralValue
+getAverageLeftWrist(dim:int):GeneralValue
```
### GeneralValue

- **valid**: boolean
- **doubleValue**: double

#### Methods

- `GeneralValue()`
- `GeneralValue(strg: String)`
- `GeneralValue(doubleValue: double)`
- `isValid(): boolean`
- `getDoubleValue(): double`
- `isLessThan(value: GeneralValue): boolean`
- `isGreaterThan(value: GeneralValue): boolean`
- `toString(): String`

#### Exception

**RuntimeException**

**InvalidValueException**

- `InvalidValueException(message: String)`
double getDoubleValue()

• InvalidValueException class: extends RunTimeException

• This method now throws InvalidValueException if this method is called on an invalid GeneralValue
  • Because this is a RunTimeException, we don’t need to explicitly declare this in the method prototype
boolean isLessThan(GeneralValue v)

Should *this* replace v as the smaller value?

<table>
<thead>
<tr>
<th><em>this</em></th>
<th>v</th>
<th>return</th>
</tr>
</thead>
<tbody>
<tr>
<td>invalid</td>
<td>invalid</td>
<td>false</td>
</tr>
<tr>
<td>invalid</td>
<td>5.7</td>
<td>false</td>
</tr>
<tr>
<td>2.38</td>
<td>invalid</td>
<td>true</td>
</tr>
<tr>
<td>2.38</td>
<td>5.7</td>
<td>true</td>
</tr>
<tr>
<td>5.7</td>
<td>2.38</td>
<td>false</td>
</tr>
<tr>
<td>5.7</td>
<td>5.7</td>
<td>false</td>
</tr>
</tbody>
</table>
Data Loading

• Trial constructor:
  • Takes as input a directory, an infant ID and a week
  • Loads the corresponding file
  • Throws FileNotFoundException if the file does not exist

• Infant constructor:
  • Takes as input a directory and an infant ID
  • Iterates through the possible weeks: 1 … MAX_WEEKS
  • Attempts to load the Trial
  • When the Trial successfully loads, add it to the list of Trials
Notes

• It is possible that all of the States in a Trial have invalid data
  • Statistics computation must acknowledge this by returning a GeneralValue and not a double

• Likewise for Trials in an Infant
Deadlines

• Project must be submitted by Wednesday, Oct 11\textsuperscript{th} @1:29pm
• Code review must be completed by Wednesday, Oct 18\textsuperscript{th}