CSE403 ● Software engineering ● sp12

Week 7-10				
Monday	Tuesday	Wednesday	Thursday	Friday
• Reading due	• Groups • Beta due		• Section	• Progress report due • Readings out
• No reading due	• Groups	• Midterm II • Reading covered [Notkin gone]	• No section	• Progress report due
Memorial Day Holiday	• Groups	• Final release due • **Project Pres. I**	• **Project Pres. II**	• **Project Pres. III**

Aspect-oriented design and programming ⇒ AspectJ

Join point

- A **join point** is a well-defined point in the control flow of a program
  - Immediately before (or after) a method execution starts (or finishes)
  - Object instantiations
  - Constructor executions
  - Field references
  - Handler executions
- Join points are execution-time notions – and each one is different even if it happens due to the exact same piece of source code in a program

Pointcut

- A **pointcut** defines – gives a name to – a set of join points
  - `execution(void Point.setX(int))`
  - `handler(ArrayOutOfBoundsException)`
  - `call(*.new(int, int))`
  - `execution(public !static * *(..))`

  -
    ```
    pointcut setter(): target(Point) &&
        (call(void setX(int)) ||
         call(void setY(int)));
    ```
- That is, pointcuts are a way to identify a subset of join points in a program execution – a cross-cutting set of join points

advice

- Join points exist
- Pointcuts are defined
- But nothing happens until some code is provided that is to execute when a join point that is part of a pointcut is reached during execution
- That code is called **advice**
- AspectJ has three kinds of advice
  - `Before`
  - `After`
  - `Around`

Hello World

- http://www.eclipse.org/ajdt/demos/HelloWorldAspectJ.html http://eclipse.org/aspectj/doc/released/progguide/starting-aspectj.html