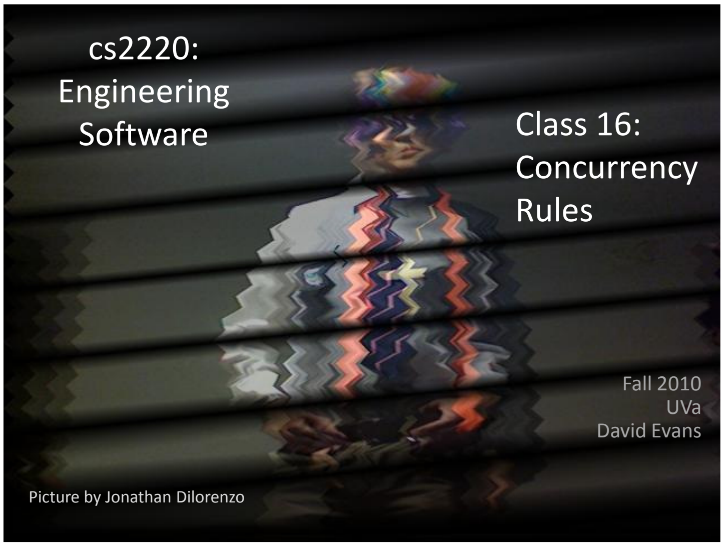


cs2220: Engineering Software

Class 16: Concurrency Rules

Fall 2010
UVa
David Evans

Picture by Jonathan Dilorenzo

Menu

- PS4
- Concurrency
- Object-Oriented Programming

PS5 Designs [Throughout the Day]

1. Blanton, James	Kalish, Michael	
2. Borja, Joseph	Oh, Uyn	Noh, Brian
3. Brown, Jeremy	Hearn, Charles	
4. Chen, Jiamin	Sparkman, Elisabeth	Sun, Yixin
5. Dewey-Vogt, Michael	Lopez, Erik	
6. Dilorenzo, Jonathan	Featherston, Joseph	
7. Dollhopf, Niklaus	Marion, John	
8. Herder, Samuel	Wallace, Alexander	

Substitution Principle

```
public class Tree {
  public Tree getChild (int n)
    // REQUIRES: 0 <= n < children.length
    // EFFECTS: Returns the Tree that is the nth leftmost child
    //                   of this.   NOTE: the rep is exposed!
```

```
public class BinaryTree extends Tree {
  // OVERVIEW: A BinaryTree is a mutable tree where the nodes are
 //    int values and each node has zero, one or two children.
  @Override
  public BinaryTree getChild (int n)
    // REQUIRES: 0 <= n < 2
    // EFFECTS: If this has at least n children, returns a copy of the
   //      BinaryTree that is the nth leftmost child of this. Otherwise,
   //      returns null.
```

Does pre_{Tree} imply $pre_{BinaryTree}$?

Substitution Principle

```
public class Tree {
  public Tree getChild (int n)
    // REQUIRES: 0 <= n < children.length
    // EFFECTS: Returns the Tree that is the nth leftmost child
    //                   of this.   NOTE: the rep is exposed!
```

```
public class BinaryTree extends Tree {
  // OVERVIEW: A BinaryTree is a mutable tree where the nodes are
 //    int values and each node has zero, one or two children.
  @Override
  public BinaryTree getChild (int n)
    // REQUIRES: 0 <= n < 2
    // EFFECTS: If this has at least n children, returns a copy of the
   //      BinaryTree that is the nth leftmost child of this. Otherwise,
   //      returns null.
```

Does $0 <= n < \text{children.length}$ imply
$0 <= n < 2$?

When children.length <= 2, yes!

```
public class Tree {
  public Tree getChild (int n)
    // REQUIRES: 0 <= n < children.length
    // EFFECTS: Returns the Tree that is the nth leftmost child
    //                   of this.   NOTE: the rep is exposed!
```

```
public class BinaryTree extends Tree {
  // OVERVIEW: A BinaryTree is a mutable tree where the nodes are
 //    int values and each node has zero, one or two children.
  @Override
  public BinaryTree getChild (int n)
    // REQUIRES: 0 <= n < 2
    // EFFECTS: If this has at least n children, returns a copy of the
   //      BinaryTree that is the nth leftmost child of this. Otherwise,
   //      returns null.
```

Does $post_{BinaryTree}$ imply $post_{Tree}$?

```
public class Tree {
  public Tree getChild (int n)
    // REQUIRES: 0 <= n < children.length
    // EFFECTS: Returns the Tree that is the nth leftmost child
    //                   of this.   NOTE: the rep is exposed!
```

```
public class BinaryTree extends Tree {
  // OVERVIEW: A BinaryTree is a mutable tree where the nodes are
 //    int values and each node has zero, one or two children.
  @Override
  public BinaryTree getChild (int n)
    // REQUIRES: 0 <= n < 2
    // EFFECTS: If this has at least n children, returns a copy of the
   //      BinaryTree that is the nth leftmost child of this. Otherwise,
   //      returns null.
```

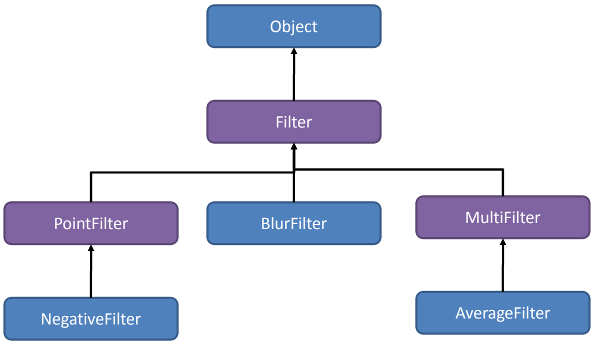
Code that “breaks” with subtype:

```
Tree b1 = t.getChild(0);
Tree b2 = t.getChild(0);
assert b1 == b2;
```

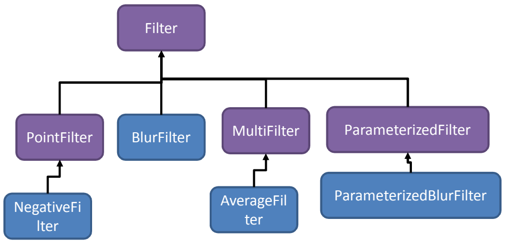
PS5 Designs [Throughout the Day]

1. Blanton, James Kalish, Michael
2. Borja, Joseph Oh, Uyn Noh, Brian
3. Brown, Jeremy Hearn, Charles
4. Chen, Jiamin Sparkman, Elisabeth Sun, Yixin
5. Dewey-Vogt, Michael Lopez, Erik
6. Dilorenzo, Jonathan Featherston, Joseph
7. Dollhopf, Niklaus Marion, John
8. Herder, Samuel Wallace, Alexander

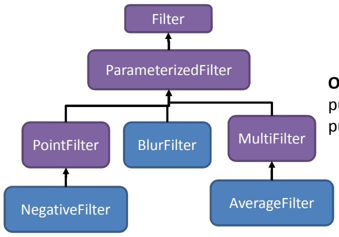
Parameterized Filters



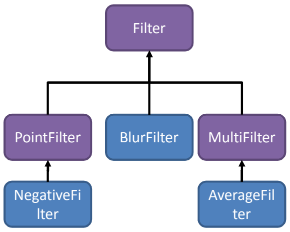
Purple: abstract classes

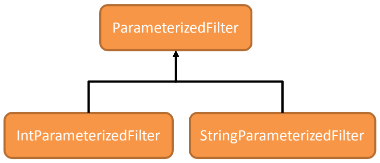


Option 1:
Filter subtype



Option 2:
public int getParameter();
public abstract boolean hasParameter();





```
public class BlurFilter extends Filter
   implements ParameterizedFilter {
  private int param;
  public void setParameter(int val) {
    param = val; }
  public int getParameter() {
    return param; }
  public String getPrompt() {
    return “Enter the blurring factor: ”; }
  …
}
```

Option 3:
Interface

```
public interface ParameterizedFilter {
   public void setParameter(int val);
   public int getParameter();
   public String getPrompt();
}
```

PS5 Designs [Throughout the Day]

1. Blanton, James Kalish, Michael
2. Borja, Joseph Oh, Uyn Noh, Brian
3. Brown, Jeremy Hearn, Charles
4. Chen, Jiamin Sparkman, Elisabeth Sun, Yixin
5. Dewey-Vogt, Michael Lopez, Erik
6. Dilorenzo, Jonathan Featherston, Joseph
7. Dollhopf, Niklaus Marion, John
8. Herder, Samuel Wallace, Alexander