

# Interfaces & Generics

---

Lecture 03, Aug 28 2007

# Administrivia: Quiz time

- Sep 6: Quiz on accessing and parsing web pages
- We'll give some examples before then, but you need to self-educate
- There will be stuff on the quiz we *haven't* covered
- Covers:
  - `java.net.URL`; accessing and downloading web pages
  - `javax.swing.text.html.HTMLEditorKit`; parsing HTML pages

# Now and then...

---

- Last time:
  - Principles of testing
  - Use of `JUnit`
  - Code demo
- This time
  - Pretest (ungraded)
  - Some statistics
  - Interfaces
  - Generics



**Pretest**

# A few statistics

---

- My implementations:
  - WebGraph.java: 175 lines
    - Compare: Graph.java: 287 lines
  - WebCrawlState.java: 125 lines
    - CrawlState.java: 175 lines
  - Crawler.java: 346 lines
    - About 1/2 in UI (main(), \_processArgs, etc.)
- Total, excluding interfaces and exceptions: 646 lines

# A few (more) statistics

---

- `TestWebGraph.java`: 490 lines
  - Supported by auxiliary object: 298 lines
- `TestCrawlState.java`: 243 lines
- Total testing code: 1031 lines
- Total object implementation code: 762 lines
  - Not counting interfaces: 300 lines

# The Interface worldview

---

- Java supports *programming by interfaces*
- Idea: don't hard-code concrete classes into your code
- Instead, refer to everything through interfaces
- Why?

# The Interface worldview

---

Design principle: **Principle of Abstraction**

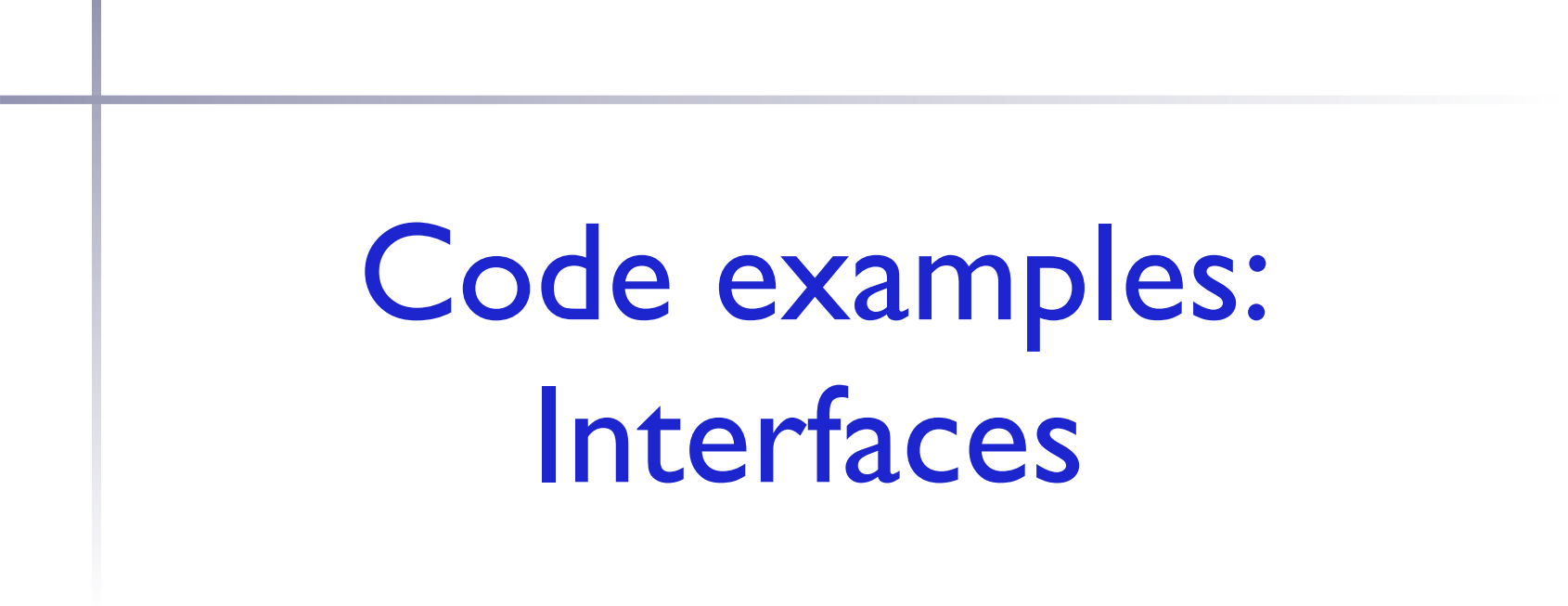
Only work with the relevant aspects of a thing;  
ignore stuff that's not immediately relevant.

# The Interface worldview

Design principle: **Centralized Definition**

Define a thing only once in a program; use a symbolic name to refer to it later.

*A.k.a.: Don't repeat yourself...*



# Code examples: Interfaces