

-ER

...

YEGOR BUGAYENKO

Lecture #5 out of 8
90 minutes

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as website. Copyright belongs to their respected authors.



Examples and Alternatives

Read and Watch



“When you need a manager, it’s often a sign that the managed are just plain old data structures and that the manager is the smart procedure doing the real work”

— Carlo Pescio

Your Coding Conventions Are Hurting You, 2011

Chapter #1:

Examples and Alternatives

[[Parser](#) Reader Controller Validator]

Parser

```
1 class Parser {  
2     static int parseInt(String t) {  
3         // Parse String into Integer  
4     }  
5     static float parseFloat(String t) {  
6         // Parse String into Float  
7     }  
8     // And many more methods...  
9 }  
10  
11 int x = Parser.parseInt("42");
```

```
1 class StringAsInt implements Number {  
2     private final String txt;  
3     StringAsInt(String t) { this.txt = t; }  
4     @Override int intValue() {  
5         // Parse String into Integer  
6         // and return the value  
7     }  
8 }  
9  
10 Number n = new StringAsInt("42");  
11 int x = n.intValue();
```

Reader

```

1 class Reader {
2     static char readChar(InputStream i) {
3         // Read the next char from the
4         // stream and return it, or NULL
5         // if the stream is at the EOF
6     }
7 }
8
9 InputStream i = new FileInputStream(..);
10 char c = Reader.readChar(i);

```

```

1 class Chars
2     private final InputStream is;
3     Chars(InputStream i)
4         this.is = i;
5     char next()
6         // Read the next char from the
7         // stream and throw exception
8         // if !exists()
9     bool exists()
10        // Return TRUE if not EOF
11
12    InputStream i = new FileInputStream(..);
13    Chars chars = new Chars(i);
14    char c = chars.next();

```

Controller

```
1 class SimpleController {
2     @GET
3     @Path("/index")
4     HttpResponse index(HttpRequest e) {
5         // Build an index page and return
6     }
7     @POST
8     @Path("/update")
9     HttpResponse update(HttpRequest e) {
10        // Save new user information
11        // and return HTTP 303
12    }
13 }
```

```
1 class IndexPage implements HttpPage
2     HttpResponse process(HttpRequest e) {
3         // Build an index page and return
4     }
5 class UpdatePage implements HttpPage
6     HttpResponse process(HttpRequest e) {
7         // Save new user information
8         // and return HTTP 303
9     }
10
11 new AllPages(
12     new IndexPage(),
13     new UpdatePage()
14 );
```

[Parser Reader Controller [Validator](#)]

Validator

```

1 class Validator {
2     bool isValid(int age) {
3         return age >= 18;
4     }
5 }
6 int a = 23;
7 Validator v = new Validator();
8 if (!v.isValid(a)) {
9     throw new Exception(
10         "Age is not valid"
11     );
12 }

```

```

1 interface Age
2     int value();
3 class DefaultAge implements Age
4     private final int a;
5     DefaultAge(int a)
6         this.a = a;
7     @Override int value()
8         return this.a;
9 class ValidAge implements Age {
10     private final Age origin;
11     ValidAge(Age age)
12         this.origin = age;
13     @Override int value()
14         int v = this.origin.value();
15         if (v < 18)
16             throw new Exception("Age is not valid");
17         return v;
18 }
19 Age a = new ValidAge(new DefaultAge(23));

```


Chapter #2:

Read and Watch

Don't Create Objects That End With -ER by me