

Java Article Center
Java Article Center

Bizcom

(Why extends is evil)

: Allen Holub
: (lemigeo@hanmail.net)
: (kkamdung@empal.com)
: Bizcom Java Article Center

Bizcom Java Article Center

2 - (ahnyounghoe@empal.com)

Allen Holub "Why extends is evil" JavaWorld

<http://www.javaworld.com/javaworld/jw-08-2003/jw-0801-toolbox.html>

"Why extends is evil" by Allen Holub was originally published by JavaWorld (www.javaworld.com), copyright JavaWorld.com, August 2003. Translated and reprinted with permission.

<http://www.javaworld.com/javaworld/jw-08-2003/jw-0801-toolbox.html>

(concrete class)

(interface)

(code)

(Summary)

80% (extends) (concrete class) (interfaces)

Gang of Four (Design Pattern)

(inheritance) (inheritance)

가

By Allen Holub

extends Charles Manson() ,

가 Gang of Four (Design Patterns)

(extends) (implements)

가 ,

가 ,

VS (Interfaces versus classes)

James Gosling () (featured speaker)

(Q&A)

가가 “ ?”

“ .” 가 ,

가 (extends)

(implements) 가

(losing flexibility)

가?

가

Agile

가

(incorporate)

(*definitely*)

가

가

가

```
f()
{
  LinkedList list = new LinkedList();
  //...
  g( list );
}
```

```
g( LinkedList list )
{
  list.add( ... );
  g2( list )
}
```

가 , LinkedList

. LinkedList HashSet 가 .

, f() Linkelist g()

, g()가 .

```
new LinkedList()   new HashSet()
(linked list)      (hash table)      가
.
.

f()
{   Collection list = new LinkedList();
    //...
    g( list );
}

g( Collection list )
{
    list.add( ... );
    g2( list )
}
.
.
```

```
f()
{   Collection c = new HashSet();
    //...
    g( c );
}

g( Collection c )
{
    for( Iterator i = c.iterator(); i.hasNext() ;)
        do_something_with( i.next() );
}
```

```

f2()
{  Collection c = new HashSet();
   //...
   g2( c.iterator() );
}

g2( Iterator i )
{  while( i.hasNext() ; )
    do_something_with( i.next() );
}
    
```

g2() (method) Map (traverse)
 , Collection (Collection derives)
 , (collection) (iterator)

(coupling)
 가 (coupling) ,
 가
 (global variable)

(coupled to the variable)
 가,
 (behavior)가
 (multithread)

가
 (OO: Object Oriented)
 . (가
). (가


```

    }

    public Object pop()
    {   return remove( --stack_pointer );
    }

    public void push_many( Object[] articles )
    {   for( int i = 0; i < articles.length; ++i )
        push( articles[i] );
    }
}

```

ArrayList clear()

```

Stack a_stack = new Stack();
a_stack.push("1");
a_stack.push("2");
a_stack.clear();

```

Stack (stack pointer)

push() 2 (stack_pointer) 가 (garbage)

Stack

ArrayList 가 , Stack (override) , removeRange()

가 ,

가 가

가 ArrayList

. , removeRange() 가 .
 . ,
 (compile-time)
 (runtime) 가 ,
 (method-not-found)
 가 ,

Stack .

```

class Stack
{
    private int stack_pointer = 0;
    private ArrayList the_data = new ArrayList();

    public void push( Object article )
    {
        the_data.add( stack_pointer++, article );
    }

    public Object pop()
    {
        return the_data.remove( --stack_pointer );
    }

    public void push_many( Object[] articles )
    {
        for( int i = 0; i < o.length; ++i )
            push( articles[i] );
    }
}
  
```

Stack

가

가

```

class Monitorable_stack extends Stack
{
    private int high_water_mark = 0;
  
```

```

private int current_size;

public void push( Object article )
{   if( ++current_size > high_water_mark )
        high_water_mark = current_size;
    super.push(article);
}

public Object pop()
{   --current_size;
    return super.pop();
}

public int maximum_size_so_far()
{   return high_water_mark;
}
}

```

, push_many()가 push()
 가
 , Stack Monitorable_stack
 push() high_water_mark가
 , 가가 , Stack 가
 , ArrayList Stack
 Stack

```

class Stack
{   private int stack_pointer = -1;
    private Object[] stack = new Object[1000];

    public void push( Object article )
    {   assert stack_pointer < stack.length;

        stack[ ++stack_pointer ] = article;
    }
}

```


, 가

.

0.1. /

```
1| import java.util.*;
2|
3| interface Stack
4| {
5|     void push( Object o );
6|     Object pop();
7|     void push_many( Object[] source );
8| }
9|
10| class Simple_stack implements Stack
11| {   private int stack_pointer = -1;
12|     private Object[] stack = new Object[1000];
13|
14|     public void push( Object o )
15|     {   assert stack_pointer < stack.length;
16|
17|         stack[ ++stack_pointer ] = o;
18|     }
19|
20|     public Object pop()
21|     {   assert stack_pointer >= 0;
22|
23|         return stack[ stack_pointer-- ];
24|     }
25|
26|     public void push_many( Object[] source )
27|     {   assert (stack_pointer + source.length) < stack.length;
28|
29|         System.arraycopy(source,0,stack,stack_pointer+1,source.length);
30|         stack_pointer += source.length;
31|     }
```

```
32| }
33|
34|
35| class Monitorable_Stack implements Stack
36| {
37|     private int high_water_mark = 0;
38|     private int current_size;
39|     Simple_stack stack = new Simple_stack();
40|
41|     public void push( Object o )
42|     {   if( ++current_size > high_water_mark )
43|         high_water_mark = current_size;
44|         stack.push(o);
45|     }
46|
47|     public Object pop()
48|     {   -- current_size;
49|         return stack.pop();
50|     }
51|
52|     public void push_many( Object[] source )
53|     {
54|         if( current_size + source.length > high_water_mark )
55|             high_water_mark = current_size + source.length;
56|
57|         stack.push_many( source );
58|     }
59|
60|     public int maximum_size()
61|     {   return high_water_mark;
62|     }
63| }
64|
```

(Frameworks)

. Microsoft Foundation Classes (MFC)

. MFC 가 , MFC

. Component paint()

. 가 MFC

. 가

. 가

. 가

(work-out-of-the-box)

(Summing up fragile base classes)

extends

implements

80%

HashMap

Map

(

“

. InputStream

InputStream

.)

가 가

, Crystal extreme programming

Agile

Gang of Four

가 가 가 가

가 Holub on Pattenrs: Learning Design Patterns by Looking at Code , 가 [Apress](http://www.apress.com) (www.apress.com)

**(Resources)**

Leonid Mikhajlov Emil Sekerinski :
<http://www.cas.mcmaster.ca/~emil/publications/fragile/ecoop98.pdf>

Allen Holub **Java Toolbox** :
<http://www.javaworld.com/columns/jw-toolbox-index.shtml>

David Geary가 **Java Design Patterns** :
<http://www.javaworld.com/columns/jw-java-design-patterns-index.shtml>

Design Patterns, Eric Gamma, Richard Helm, Ralph Johnson, John Vlissides
 (Addison - Wesley Publishing Co., 1995; ISBN: 0201633612):
<http://www.amazon.com/exec/obidos/ASIN/0201633612/javaworld>

JavaWorld Topical Index **Design Patterns:**
http://www.javaworld.com/channel_content/jw-patterns-index.shtml

JavaWorld Topical Index **Object-Oriented Design and Programming:**
http://www.javaworld.com/channel_content/jw-oop-index.shtml

JavaWorld
<http://www.javaworld.com/javaforums/ubbthreads.php?Cat=&C=2>

JavaWorld 가
<http://www.javaworld.com/subscribe>