Method Types

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

ADAP C01

Method Types

- A method type classifies a method into a particular type
  - The method type is indicative of the main purpose
  - A method may have only one type, not many
  - Thus, a method should have one purpose
- Different method types are orthogonal to each other
  - **Query methods**
  - **Mutation methods**
  - **Helper methods**
- A method type comes with its own conventions
  - Naming conventions, for example, specific leading verbs
  - Specific implementation structures

Main Categories of Method Types

- **Query methods**
  - Methods that return information about the object but don’t change its state
- **Mutation methods**
  - Methods that change the object’s state but don’t provide information back
- **Helper methods**
  - Methods that perform some utility function independent of the object

Categories and Examples of Method Types

Query Method	Mutation Method	Helper Method
get method (getter)	set method (setter)	factory method
boolean query method	command method	cloning method
comparison method	initialization method	assertion method
conversion method	finalization method	logging method
...	...	...

A Simple Class for Homogenous Names

- Homogenous name
  - A multi-component text string
    - of same-type components with
    - a single delimiter character
- Homogenous name examples
  - Domain names (“www.jvalue.org”)
  - Java path names (“org.jvalue.names.Name”)
  - Directory paths (“/home/dirk/docs”)
- Unlike heterogeneous names
  - “http://www.jvalue.org/index.html”

General Method Naming Rule

verb

(+ optional noun)

then, be as specific as possible

Get Method (Query Method)

Definition	A get method is a query method that returns a (logical) field of the object.
Also known as	Getter
JDK example	Class Object#getClass() Object Enumeration#nextElement()
Name example	String getComponent(int) Iterable<String> getComponentIterator()
Prefixes	get
Naming	After the prefix, the name of the field being queried follows.

Get Method Examples

```
public class Name {

  protected String name;
  protected int noComponents;

  public int getNoComponents() {
    return noComponents;
  }

  public String getComponent(int index) {
    assertIsValidIndex(index);
    return doGetComponent(index);
  }

  protected String doGetComponent(int i) {
    int startPos = getStartPositionOfComponent(i);
    int endPos = getEndPositionOfComponent(i);
    String maskedComponent = name.substring(startPos, endPos);
    return NameHelper.unmaskString(maskedComponent);
  }

  ...
}
```

Boolean Query Method (Query Method)

Definition	A boolean query method is a query method that returns a boolean value.
Also known as	-
JDK example	boolean Object#equals()
Name example	boolean isEmpty()
Prefixes	is, has, may, can, …
Naming	After the prefix, the aspect being queried follows.

Boolean Query Method Examples

```
public boolean hasComponent(int index) {
  return doHasComponent(index);
}

protected boolean doHasComponent(int index) {
  return (0 <= index) && (index < getNoComponents());
}

public boolean isEmpty() {
  return getNoComponents() == 0;
}
```

Comparison Method (Query Method)

Definition	A comparison method is a query method that compares two objects using an ordinal scale.
Also known as	Comparing method
JDK example	int Comparable#compareTo(Object)
Name example	-
Prefixes	-
Naming	-

Comparison Method Examples

```
public int compareTo(Integer anotherInteger) {
  int thisVal = this.value;
  int anotherVal = anotherInteger.value;
  if (thisVal < anotherVal) {
    return -1;
  } else if (thisVal == anotherVal) {
    return 0;
  } else {
    return 1;
  }
}
```

Conversion Method (Query Method)

Definition	A conversion method is a query method that returns a different representation of this object.
Also known as	Converter method, interpretation method
JDK example	String Object#toString() int Integer#intValue()
Name example	String asString() String asStringWith(char)
Prefixes	as, to
Naming	After the prefix, typically the class name being converted to follows.

Conversion Method Examples

```
public String[] asStringArray() {
  int max = getNoComponents();
  String[] sa = new String[max];
  for (int i = 0; i < max; i++) {
    sa[i] = getComponent(i);
  }

  return sa;
}
```

Set Method (Mutation Method)

Definition	A set method is a mutation method that sets a (logical) field of an object to some value.
Also known as	Setter
JDK example	void Thread#setPriority(int)
Name example	void Name#setComponent(int, String)
Prefixes	set
Naming	After the prefix (if any), the field being changed follows.

Set Method Examples

```
public void setComponent(int i, String c) {
  assertIsValidIndex(i);
  assertIsNonNullArgument(c);
  int oldNoComponents = getNoComponents();

  doSetComponent(i, c);

  assert c.equals(getComponent(i)) : "postcondition failed";
  assert oldNoComponents == getNoComponents() : "pc failed";
}

protected void doSetComponent(int i, String c) {
  doInsert(i, c);
  doRemove(i + 1);
}
```

Command Method (Mutation Method)

Definition	A command method is a method that executes a complex change to the object's state.
Also known as	-
JDK example	void Object#notify() void JComponent#repaint()
Name example	void insert(int i, String c) void remove(int i)
Prefixes	handle, execute, make
Naming	-

Command Method Examples

```
public void insert(int i, String c) {
  assertIsValidIndex(i, getNoComponents() + 1);
  assertIsNonNullArgument(c);
  int oldNoComponents = getNoComponents();

  doInsert(i, c);

  assert (oldNoComponents + 1) == getNoComponents() : "pc failed";
}

protected void doInsert(int index, String component) {
  int newSize = getNoComponents() + 1;
  String[] newComponents = new String[newSize];
  for (int i = 0, j = 0; j < newSize; j++) {
    if (j != index) {
      newComponents[j] = components[i++];
    } else {
      newComponents[j] = component;
    }
  }
  components = newComponents;
}
```

Initialization Method (Mutation Method)

Definition	An initialization method is a mutation method that sets some or all fields of an object to an initial value.
Also known as	-
JDK example	void LookAndFeel#initialize()
Name example	-
Prefixes	init, initialize
Naming	If prefixed with init, typically the name of the object part being initialized follows.

Initialization Method Examples

```
public class NameTest {

  protected Name defaultName;
  protected Name emptyDefaultName;
  protected Name compactName;
  protected Name emptyCompactName;

  protected void initNames(String[] arg) {
    defaultName = new StringArrayName(arg);
    compactName = new StringName(arg);
  }

  protected void initEmptyNames() {
    emptyDefaultName = new StringArrayName();
    emptyCompactName = new StringName();
  }

  ...

}
```

Factory Method (Helper Method)

Definition	A factory method is a helper method that creates an object and returns it to the client.
Also known as	Object creation method
JDK example	String String#valueOf(int) String String#valueOf(double)
Name example	-
Prefixes	create, make, build, (new, getNew)
Naming	After the prefix, the product name follows.

Factory Method Example

```
public class PhotoManager extends ObjectManager {

  protected static final PhotoManager instance = new PhotoManager();

  ...

  public Photo createPhoto(String fn, Image ui) throws Exception {
    PhotoId id = PhotoId.getNextId();
    Photo result = PhotoUtil.createPhoto(fn, id, ui);
    addPhoto(result);
    return result;
  }

  ...
}
```

Assertion Method (Helper Method)

Definition	An assertion method is a helper method that tests a condition. If the condition holds, it returns silently. If it does not, an exception is thrown.
Also known as	-
JDK example	void AccessControlContext#checkPermission(Permission) throws AccessControlException
Name example	void Name#assertIsValidIndex(int) throws IndexOutOfBoundsException
Prefixes	assert, check, test
Naming	After the prefix, the condition being checked follows.

Assertion Method Examples

```
protected void assertIsValidIndex(int i)
  throws IndexOutOfBoundsException {
  assertIsValidIndex(i, getNoComponents());
}

protected void assertIsValidIndex(int i, int upperLimit)
  throws IndexOutOfBoundsException {
  if ((i < 0) || (i >= upperLimit)) {
    throw new IndexOutOfBoundsException("invalid index = " + i);
  }
}

protected void assertIsNonNullArgument(Object o) {
  if (o == null) {
    throw new IllegalArgumentException("received null argument");
  }
}
```

Quiz: java.lang.Object Method Names

1. How do you classify these methods?
   - protected Object clone()
   - boolean equals(Object obj)
   - protected void finalize()
   - Class<?> getClass()
   - int hashCode()
   - void notify()
   - void notifyAll()
   - String toString()
   - void wait()
   - void wait(long timeout)
   - void wait(long timeout, int nanos)

More Method Types

- Method type hierarchies can be much deeper, for example:
  - Object creation method
    - Cloning method: Clone object at hand
    - Factory method: Create related object
      - Copying method: Create by copying argument
      - Trading method: Create by resolving specification
    - ...

Naming Object Creation Methods

- Creation methods prefixed with “create”
  - Should create the object (sans abnormal situation)
- Creation methods prefixed with “ensure”
  - Should ensure a particular object exists, possibly creating it

Single-Purpose Rule

- Definition of single-purpose rule
  - A method should serve one main purpose
  - Derives from single method-type rule
- Benefit of single-purpose rule
  - Make the method more easy to understand
  - Makes overriding methods easier

Exceptions to the Single-Purpose Rule

- Critical sections
  - Increment and return
    - Special case: Iterators
- Lazy initialization

Making Method Types Explicit in Code

- Annotate (in comments) using @MethodType method-type

Quiz: Naming a Factory Method

1. You need a method to create a new photo. Which name is best?
    1. PhotoFactory.make()
    2. PhotoFactory.newPhoto()
    3. PhotoFactory.createPhoto()
    4. PhotoFactory.createNewPhoto()

Review / Summary of Session

- General method types
  - What are method types?
  - What categories of method types are there?
- Specific method types
  - What specific method types are there? How common are they?
  - How are they defined? What naming convention do they follow?
- Exceptions to the rules
  - Are there exceptions to the rules? What are they?
  - How are they justified? What are programming idioms?

Thank you! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

Credits and License

- Original version
  - © 2012-2019 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...

Method Types

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

ADAP C01

It is Friedrich-Alexander University Erlangen-Nürnberg – FAU, in short.

Corporate identity wants us to say “Friedrich-Alexander University”.

Method Types

- A method type classifies a method into a particular type
  - The method type is indicative of the main purpose
  - A method may have only one type, not many
  - Thus, a method should have one purpose
- Different method types are orthogonal to each other
  - **Query methods**
  - **Mutation methods**
  - **Helper methods**
- A method type comes with its own conventions
  - Naming conventions, for example, specific leading verbs
  - Specific implementation structures

Main Categories of Method Types

- **Query methods**
  - Methods that return information about the object but don’t change its state
- **Mutation methods**
  - Methods that change the object’s state but don’t provide information back
- **Helper methods**
  - Methods that perform some utility function independent of the object

Categories and Examples of Method Types

Query Method	Mutation Method	Helper Method
get method (getter)	set method (setter)	factory method
boolean query method	command method	cloning method
comparison method	initialization method	assertion method
conversion method	finalization method	logging method
...	...	...

A Simple Class for Homogenous Names

- Homogenous name
  - A multi-component text string
    - of same-type components with
    - a single delimiter character
- Homogenous name examples
  - Domain names (“www.jvalue.org”)
  - Java path names (“org.jvalue.names.Name”)
  - Directory paths (“/home/dirk/docs”)
- Unlike heterogeneous names
  - “http://www.jvalue.org/index.html”

General Method Naming Rule

verb

(+ optional noun)

then, be as specific as possible

Get Method (Query Method)

Definition	A get method is a query method that returns a (logical) field of the object.
Also known as	Getter
JDK example	Class Object#getClass() Object Enumeration#nextElement()
Name example	String getComponent(int) Iterable<String> getComponentIterator()
Prefixes	get
Naming	After the prefix, the name of the field being queried follows.

Get Method Examples

```
public class Name {

  protected String name;
  protected int noComponents;

  public int getNoComponents() {
    return noComponents;
  }

  public String getComponent(int index) {
    assertIsValidIndex(index);
    return doGetComponent(index);
  }

  protected String doGetComponent(int i) {
    int startPos = getStartPositionOfComponent(i);
    int endPos = getEndPositionOfComponent(i);
    String maskedComponent = name.substring(startPos, endPos);
    return NameHelper.unmaskString(maskedComponent);
  }

  ...
}
```

Boolean Query Method (Query Method)

Definition	A boolean query method is a query method that returns a boolean value.
Also known as	-
JDK example	boolean Object#equals()
Name example	boolean isEmpty()
Prefixes	is, has, may, can, …
Naming	After the prefix, the aspect being queried follows.

Boolean Query Method Examples

```
public boolean hasComponent(int index) {
  return doHasComponent(index);
}

protected boolean doHasComponent(int index) {
  return (0 <= index) && (index < getNoComponents());
}

public boolean isEmpty() {
  return getNoComponents() == 0;
}
```

Comparison Method (Query Method)

Definition	A comparison method is a query method that compares two objects using an ordinal scale.
Also known as	Comparing method
JDK example	int Comparable#compareTo(Object)
Name example	-
Prefixes	-
Naming	-

Comparison Method Examples

```
public int compareTo(Integer anotherInteger) {
  int thisVal = this.value;
  int anotherVal = anotherInteger.value;
  if (thisVal < anotherVal) {
    return -1;
  } else if (thisVal == anotherVal) {
    return 0;
  } else {
    return 1;
  }
}
```

Conversion Method (Query Method)

Definition	A conversion method is a query method that returns a different representation of this object.
Also known as	Converter method, interpretation method
JDK example	String Object#toString() int Integer#intValue()
Name example	String asString() String asStringWith(char)
Prefixes	as, to
Naming	After the prefix, typically the class name being converted to follows.

Conversion Method Examples

```
public String[] asStringArray() {
  int max = getNoComponents();
  String[] sa = new String[max];
  for (int i = 0; i < max; i++) {
    sa[i] = getComponent(i);
  }

  return sa;
}
```

Set Method (Mutation Method)

Definition	A set method is a mutation method that sets a (logical) field of an object to some value.
Also known as	Setter
JDK example	void Thread#setPriority(int)
Name example	void Name#setComponent(int, String)
Prefixes	set
Naming	After the prefix (if any), the field being changed follows.

Set Method Examples

```
public void setComponent(int i, String c) {
  assertIsValidIndex(i);
  assertIsNonNullArgument(c);
  int oldNoComponents = getNoComponents();

  doSetComponent(i, c);

  assert c.equals(getComponent(i)) : "postcondition failed";
  assert oldNoComponents == getNoComponents() : "pc failed";
}

protected void doSetComponent(int i, String c) {
  doInsert(i, c);
  doRemove(i + 1);
}
```

Command Method (Mutation Method)

Definition	A command method is a method that executes a complex change to the object's state.
Also known as	-
JDK example	void Object#notify() void JComponent#repaint()
Name example	void insert(int i, String c) void remove(int i)
Prefixes	handle, execute, make
Naming	-

Command Method Examples

```
public void insert(int i, String c) {
  assertIsValidIndex(i, getNoComponents() + 1);
  assertIsNonNullArgument(c);
  int oldNoComponents = getNoComponents();

  doInsert(i, c);

  assert (oldNoComponents + 1) == getNoComponents() : "pc failed";
}

protected void doInsert(int index, String component) {
  int newSize = getNoComponents() + 1;
  String[] newComponents = new String[newSize];
  for (int i = 0, j = 0; j < newSize; j++) {
    if (j != index) {
      newComponents[j] = components[i++];
    } else {
      newComponents[j] = component;
    }
  }
  components = newComponents;
}
```

Initialization Method (Mutation Method)

Definition	An initialization method is a mutation method that sets some or all fields of an object to an initial value.
Also known as	-
JDK example	void LookAndFeel#initialize()
Name example	-
Prefixes	init, initialize
Naming	If prefixed with init, typically the name of the object part being initialized follows.

Initialization Method Examples

```
public class NameTest {

  protected Name defaultName;
  protected Name emptyDefaultName;
  protected Name compactName;
  protected Name emptyCompactName;

  protected void initNames(String[] arg) {
    defaultName = new StringArrayName(arg);
    compactName = new StringName(arg);
  }

  protected void initEmptyNames() {
    emptyDefaultName = new StringArrayName();
    emptyCompactName = new StringName();
  }

  ...

}
```

Factory Method (Helper Method)

Definition	A factory method is a helper method that creates an object and returns it to the client.
Also known as	Object creation method
JDK example	String String#valueOf(int) String String#valueOf(double)
Name example	-
Prefixes	create, make, build, (new, getNew)
Naming	After the prefix, the product name follows.

Factory Method Example

```
public class PhotoManager extends ObjectManager {

  protected static final PhotoManager instance = new PhotoManager();

  ...

  public Photo createPhoto(String fn, Image ui) throws Exception {
    PhotoId id = PhotoId.getNextId();
    Photo result = PhotoUtil.createPhoto(fn, id, ui);
    addPhoto(result);
    return result;
  }

  ...
}
```

Assertion Method (Helper Method)

Definition	An assertion method is a helper method that tests a condition. If the condition holds, it returns silently. If it does not, an exception is thrown.
Also known as	-
JDK example	void AccessControlContext#checkPermission(Permission) throws AccessControlException
Name example	void Name#assertIsValidIndex(int) throws IndexOutOfBoundsException
Prefixes	assert, check, test
Naming	After the prefix, the condition being checked follows.

Assertion Method Examples

```
protected void assertIsValidIndex(int i)
  throws IndexOutOfBoundsException {
  assertIsValidIndex(i, getNoComponents());
}

protected void assertIsValidIndex(int i, int upperLimit)
  throws IndexOutOfBoundsException {
  if ((i < 0) || (i >= upperLimit)) {
    throw new IndexOutOfBoundsException("invalid index = " + i);
  }
}

protected void assertIsNonNullArgument(Object o) {
  if (o == null) {
    throw new IllegalArgumentException("received null argument");
  }
}
```

Quiz: java.lang.Object Method Names

1. How do you classify these methods?
   - protected Object clone()
   - boolean equals(Object obj)
   - protected void finalize()
   - Class<?> getClass()
   - int hashCode()
   - void notify()
   - void notifyAll()
   - String toString()
   - void wait()
   - void wait(long timeout)
   - void wait(long timeout, int nanos)

More Method Types

- Method type hierarchies can be much deeper, for example:
  - Object creation method
    - Cloning method: Clone object at hand
    - Factory method: Create related object
      - Copying method: Create by copying argument
      - Trading method: Create by resolving specification
    - ...

Naming Object Creation Methods

- Creation methods prefixed with “create”
  - Should create the object (sans abnormal situation)
- Creation methods prefixed with “ensure”
  - Should ensure a particular object exists, possibly creating it

Single-Purpose Rule

- Definition of single-purpose rule
  - A method should serve one main purpose
  - Derives from single method-type rule
- Benefit of single-purpose rule
  - Make the method more easy to understand
  - Makes overriding methods easier

Exceptions to the Single-Purpose Rule

- Critical sections
  - Increment and return
    - Special case: Iterators
- Lazy initialization

Making Method Types Explicit in Code

- Annotate (in comments) using @MethodType method-type

Quiz: Naming a Factory Method

1. You need a method to create a new photo. Which name is best?
    1. PhotoFactory.make()
    2. PhotoFactory.newPhoto()
    3. PhotoFactory.createPhoto()
    4. PhotoFactory.createNewPhoto()

Review / Summary of Session

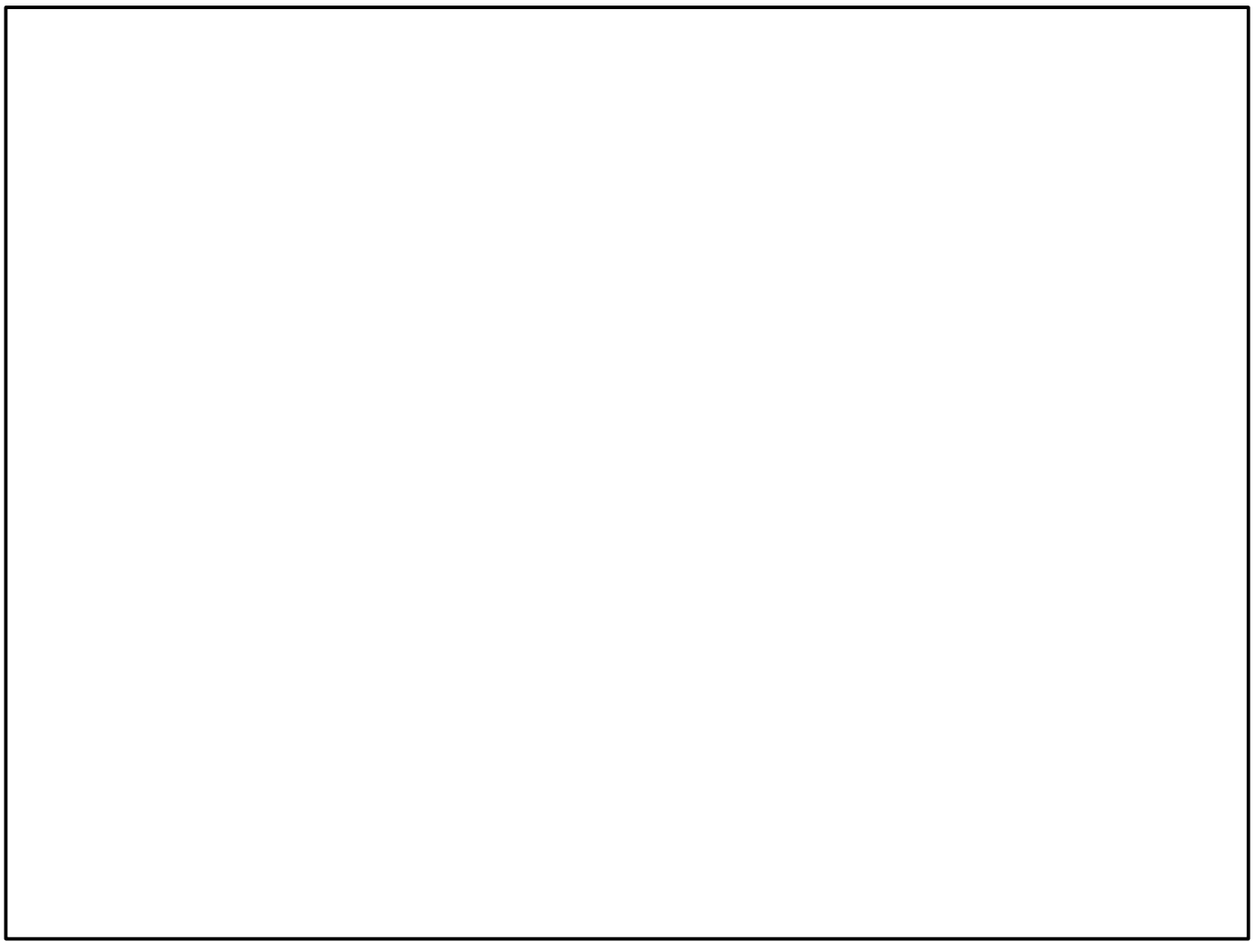
- General method types
  - What are method types?
  - What categories of method types are there?
- Specific method types
  - What specific method types are there? How common are they?
  - How are they defined? What naming convention do they follow?
- Exceptions to the rules
  - Are there exceptions to the rules? What are they?
  - How are they justified? What are programming idioms?

Thank you! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

DR



Credits and License

- Original version
  - © 2012-2019 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...