Overview Package Class Tree Deprecated Index Help

PREV CLASS NEXT CLASS | FRAMES NO FRAMES All Classes All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD | DETAIL: FIELD | CONSTR | METHOD

org.aspectj.lang

Interface JoinPoint

11 - Sep - 2012 - initiated & completed.

All Known Subinterfaces:

ProceedingJoinPoint

`public interface` **`JoinPoint`**

Provides reflective access to both the state available at a join point and static information about it. This information is available from the body of advice using the special form `thisJoinPoint`. The primary use of this reflective information is for tracing and logging applications.

```
 aspect Logging {
     before(): within(com.bigboxco..*) && execution(public * *(..)) {
         System.err.println("entering: " + thisJoinPoint);
         System.err.println("  w/args: " + thisJoinPoint.getArgs());
         System.err.println("      at: " + thisJoinPoint.getSourceLocation());
     }
 }
```

Nested Class Summary

static interface	**JoinPoint.EnclosingStaticPart**
static interface	**JoinPoint.StaticPart** This helper object contains only the static information about a join point.

Field Summary

static java.lang.String	**ADVICE_EXECUTION**
static java.lang.String	**CONSTRUCTOR_CALL**
static java.lang.String	**CONSTRUCTOR_EXECUTION**
static java.lang.String	**EXCEPTION_HANDLER**
static java.lang.String	**FIELD_GET**
static java.lang.String	**FIELD_SET**
static java.lang.String	**INITIALIZATION**
static java.lang.String	

	METHOD_CALL
static java.lang.String	**METHOD_EXECUTION** The legal return values from getKind()
static java.lang.String	**PREINITIALIZATION**
static java.lang.String	**STATICINITIALIZATION**
static java.lang.String	**SYNCHRONIZATION_LOCK**
static java.lang.String	**SYNCHRONIZATION_UNLOCK**

Method Summary

java.lang.Object[]	**getArgs**() Returns the arguments at this join point.
java.lang.String	**getKind**() Returns a String representing the kind of join point. method-execution
Signature	**getSignature**() Returns the signature at the join point.
SourceLocation	**getSourceLocation**() Returns the source location corresponding to the join point.
JoinPoint.StaticPart	**getStaticPart**() Returns an object that encapsulates the static parts of this join point.
java.lang.Object	**getTarget**() Returns the target object. both method return same object.
java.lang.Object	**getThis**() Returns the currently executing object.
java.lang.String	**toLongString**() Returns an extended string representation of the join point.
java.lang.String	**toShortString**() Returns an abbreviated string representation of the join point.
java.lang.String	**toString**()

these two return pointcut expression

Field Detail

METHOD_EXECUTION

```
static final java.lang.String METHOD_EXECUTION
```

The legal return values from getKind()

See Also:

[Constant Field Values](#)

METHOD_CALL

```
static final java.lang.String METHOD_CALL
```

See Also:
[Constant Field Values](#)

CONSTRUCTOR_EXECUTION

```
static final java.lang.String CONSTRUCTOR_EXECUTION
```

See Also:
[Constant Field Values](#)

CONSTRUCTOR_CALL

```
static final java.lang.String CONSTRUCTOR_CALL
```

See Also:
[Constant Field Values](#)

FIELD_GET

```
static final java.lang.String FIELD_GET
```

See Also:
[Constant Field Values](#)

FIELD_SET

```
static final java.lang.String FIELD_SET
```

See Also:
[Constant Field Values](#)

STATICINITIALIZATION

```
static final java.lang.String STATICINITIALIZATION
```

See Also:
[Constant Field Values](#)

PREINITIALIZATION

```
static final java.lang.String PREINITIALIZATION
```

See Also:
Constant Field Values

INITIALIZATION

```
static final java.lang.String INITIALIZATION
```

See Also:
Constant Field Values

EXCEPTION_HANDLER

```
static final java.lang.String EXCEPTION_HANDLER
```

See Also:
Constant Field Values

SYNCHRONIZATION_LOCK

```
static final java.lang.String SYNCHRONIZATION_LOCK
```

See Also:
Constant Field Values

SYNCHRONIZATION_UNLOCK

```
static final java.lang.String SYNCHRONIZATION_UNLOCK
```

See Also:
Constant Field Values

ADVICE_EXECUTION

```
static final java.lang.String ADVICE_EXECUTION
```

See Also:
Constant Field Values

Method Detail

toString

```
java.lang.String toString()
```

Overrides:
`toString` in class `java.lang.Object`

toShortString

```
java.lang.String toShortString()
```

Returns an abbreviated string representation of the join point.

toLongString

```
java.lang.String toLongString()
```

Returns an extended string representation of the join point.

getThis

```
java.lang.Object getThis()
```

Returns the currently executing object. This will always be the same object as that matched by the `this` pointcut designator. Unless you specifically need this reflective access, you should use the `this` pointcut designator to get at this object for better static typing and performance.

Returns null when there is no currently executing object available. This includes all join points that occur in a static context.

getTarget

```
java.lang.Object getTarget()
```

Returns the target object. This will always be the same object as that matched by the `target` pointcut designator. Unless you specifically need this reflective access, you should use the `target` pointcut designator to get at this object for better static typing and performance.

Returns null when there is no target object.

getArgs

```
java.lang.Object[] getArgs()
```

Returns the arguments at this join point.

getSignature

```
Signature getSignature()
```

Returns the signature at the join point. `getStaticPart().getSignature()` returns the same object

getSourceLocation

Spring AOP does not support line no, file name

```
SourceLocation getSourceLocation()
```

Returns the source location corresponding to the join point.

If there is no source location available, returns null.

Returns the SourceLocation of the defining class for default constructors.

`getStaticPart().getSourceLocation()` returns the same object.

---

getKind

`java.lang.String` **`getKind`**`()`

Returns a String representing the kind of join point. This String is guaranteed to be interned. `getStaticPart().getKind()` returns the same object.

---

getStaticPart

[`JoinPoint.StaticPart`] **`getStaticPart`**`()`

Returns an object that encapsulates the static parts of this join point.

Overview **Package** **Class** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview **Package** **Class** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

org.aspectj.lang

Interface ProceedingJoinPoint

All Superinterfaces:

JoinPoint

```
public interface ProceedingJoinPoint
extends JoinPoint
```

ProceedingJoinPoint exposes the proceed(..) method in order to support around advice in @AJ aspects

Author:

Alexandre Vasseur

Nested Class Summary

Nested classes/interfaces inherited from interface org.aspectj.lang.JoinPoint
`JoinPoint.EnclosingStaticPart`, `JoinPoint.StaticPart`

Field Summary

Fields inherited from interface org.aspectj.lang.JoinPoint
`ADVICE_EXECUTION`, `CONSTRUCTOR_CALL`, `CONSTRUCTOR_EXECUTION`, `EXCEPTION_HANDLER`, `FIELD_GET`, `FIELD_SET`, `INITIALIZATION`, `METHOD_CALL`, `METHOD_EXECUTION`, `PREINITIALIZATION`, `STATICINITIALIZATION`, `SYNCHRONIZATION_LOCK`, `SYNCHRONIZATION_UNLOCK`

Method Summary

`java.lang.Object`	**`proceed`**`()` Proceed with the next advice or target method invocation
`java.lang.Object`	**`proceed`**`(java.lang.Object[] args)` Proceed with the next advice or target method invocation
`void`	**`set$AroundClosure`**`(org.aspectj.runtime.internal.AroundClosure arc)` The joinpoint needs to know about its closure so that proceed can delegate to closure.run() This internal method should not be called directly, and won't be visible to the end-user when packed in a jar (synthetic method)

Methods inherited from interface org.aspectj.lang.JoinPoint
`getArgs`, `getKind`, `getSignature`, `getSourceLocation`, `getStaticPart`, `getTarget`, `getThis`,

toLongString, toShortString, toString

Method Detail

set$AroundClosure

```
void set$AroundClosure(org.aspectj.runtime.internal.AroundClosure arc)
```

The joinpoint needs to know about its closure so that proceed can delegate to closure.run()

This internal method should not be called directly, and won't be visible to the end-user when packed in a jar (synthetic method)

Parameters:
`arc` -

---

proceed

```
java.lang.Object proceed()
                         throws java.lang.Throwable
```

Proceed with the next advice or target method invocation

Returns:
Throws:
`java.lang.Throwable`

---

proceed

```
java.lang.Object proceed(java.lang.Object[] args)
                         throws java.lang.Throwable
```

Proceed with the next advice or target method invocation

Unlike code style, proceed(..) in annotation style places different requirements on the parameters passed to it. The proceed(..) call takes, in this order:

- If 'this()' was used in the pointcut for binding, it must be passed first in proceed(..).
- If 'target()' was used in the pointcut for binding, it must be passed next in proceed(..) - it will be the first argument to proceed(..) if this() was not used for binding.
- Finally come all the arguments expected at the join point, in the order they are supplied at the join point. Effectively the advice signature is ignored - it doesn't matter if a subset of arguments were bound or the ordering was changed in the advice signature, the proceed(..) calls takes all of them in the right order for the join point.

Since proceed(..) in this case takes an Object array, AspectJ cannot do as much compile time checking as it can for code style. If the rules above aren't obeyed then it will unfortunately manifest as a runtime error.

Parameters:
`args` -
Returns:
Throws:

```
            java.lang.Throwable
```

Overview Package Class Tree Deprecated Index Help
PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

org.aspectj.lang

Interface Signature

All Known Subinterfaces:

AdviceSignature, CatchClauseSignature, CodeSignature, ConstructorSignature, FieldSignature, InitializerSignature, LockSignature, MemberSignature, MethodSignature, UnlockSignature

All Known Implementing Classes:

FieldSignatureImpl

```
public interface Signature
```

Represents the signature at a join point. This interface parallels `java.lang.reflect.Member`.

This interface is typically used for tracing or logging applications to obtain reflective information about the join point, i.e. using the j2se 1.4 `java.util.logging` API

```
 aspect Logging {
     Logger logger = Logger.getLogger("MethodEntries");

     before(): within(com.bigboxco..*) && execution(public * *(..)) {
         Signature sig = thisJoinPoint.getSignature();
         logger.entering(sig.getDeclaringType().getName(),
                         sig.getName());
     }
 }
```

More detailed information about a specific kind of signature can be obtained by casting this `Signature` object into one of its more specific sub-types available in `org.aspectj.lang.reflect`.

See Also:

`Member`, `java.util.logging.Logger`

Method Summary

`java.lang.Class`	**`getDeclaringType`**`()` Returns a `java.lang.Class` object representing the class, interface, or aspect that declared this member.
`java.lang.String`	**`getDeclaringTypeName`**`()` Returns the fully-qualified name of the declaring type.
`int`	**`getModifiers`**`()` Returns the modifiers on this signature represented as an int.
`java.lang.String`	**`getName`**`()` Returns the identifier part of this signature.
`java.lang.String`	**`toLongString`**`()` Returns an extended string representation of this signature.
`java.lang.String`	`()`

	toShortString Returns an abbreviated string representation of this signature.
`java.lang.String`	**toString**()

Method Detail

toString

`java.lang.String` **`toString`**`()`

Overrides:
`toString` in class `java.lang.Object`

---

toShortString

`java.lang.String` **`toShortString`**`()`

Returns an abbreviated string representation of this signature.

---

toLongString

`java.lang.String` **`toLongString`**`()`

Returns an extended string representation of this signature.

---

getName

`java.lang.String` **`getName`**`()`

Returns the identifier part of this signature. For methods this will return the method name.

See Also:
`Member.getName()`

i will use this for logging

---

getModifiers

`int` **`getModifiers`**`()`

Returns the modifiers on this signature represented as an int. Use the constants and helper methods defined on `java.lang.reflect.Modifier` to manipulate this, i.e.

```
// check if this signature is public
java.lang.reflect.Modifier.isPublic(sig.getModifiers());

// print out the modifiers
java.lang.reflect.Modifier.toString(sig.getModifiers());
```

See Also:

`Member.getModifiers()`, `Modifier`

---

getDeclaringType

dont use this for simply get class name for logging.

```
java.lang.Class getDeclaringType()
```

Returns a `java.lang.Class` object representing the class, interface, or aspect that declared this member. For intra-member declarations, this will be the type on which the member is declared, not the type where the declaration is lexically written. Use `SourceLocation.getWithinType()` to get the type in which the declaration occurs lexically.

For consistency with `java.lang.reflect.Member`, this method should have been named `getDeclaringClass()`.

See Also:

`Member.getDeclaringClass()`

---

getDeclaringTypeName

use this... to get class name for logging

```
java.lang.String getDeclaringTypeName()
```

Returns the fully-qualified name of the declaring type. This is equivalent to calling getDeclaringType().getName(), but caches the result for greater efficiency.

Overview **Package** **Class** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

---

11 - sep - 2012