

# Programowanie aspektowe na przykładzie AspectJ

# Plan prezentacji

- Co to programowanie aspektowe i AspectJ
- Szybki start z AspectJ (przykład)
- Uruchamianie aplikacji z aspektami
- Jak to działa
- Możliwości wplatania kodu
- Przykład
- Odpowiedzi na pytania

# Programowanie aspektowe i AspectJ

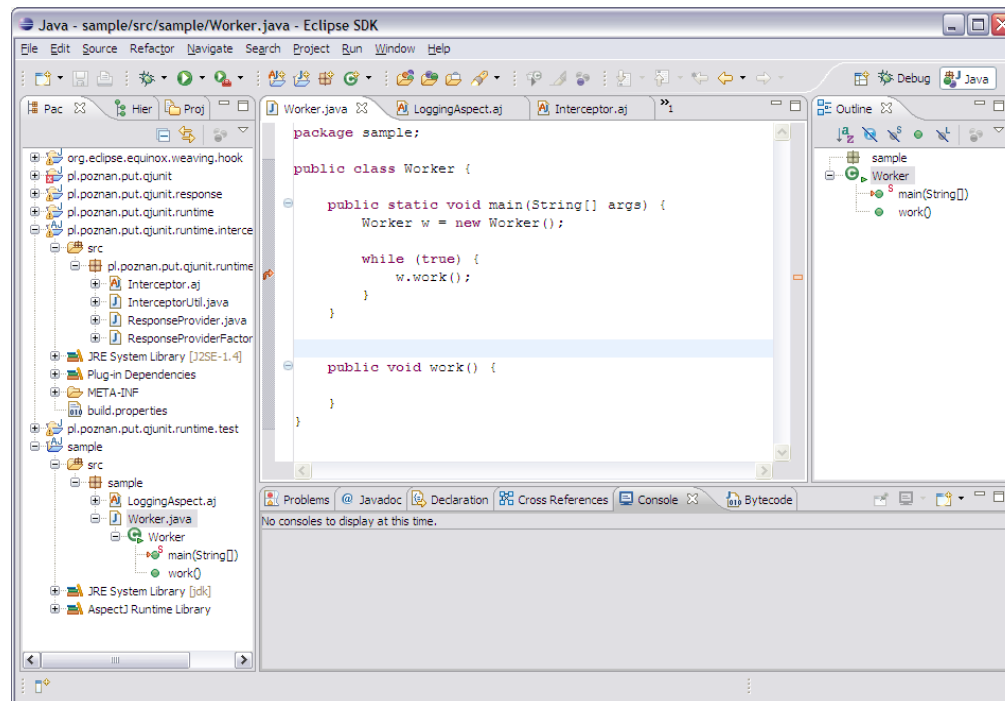
- AOP = Aspect-oriented Programming
- Rozwiązywanie zagadnień wspólnych dla wielu niezależnych komponentów/modułów projektu.
- Przeciwiństwo innych metodyk, nastawionych na modelowanie systemu (jako moduły, komponenty).
- AspectJ: realizacja koncepcji programowania aspektowego dla języka Java. Prawdopodobnie najpopularniejsza

# AspectJ

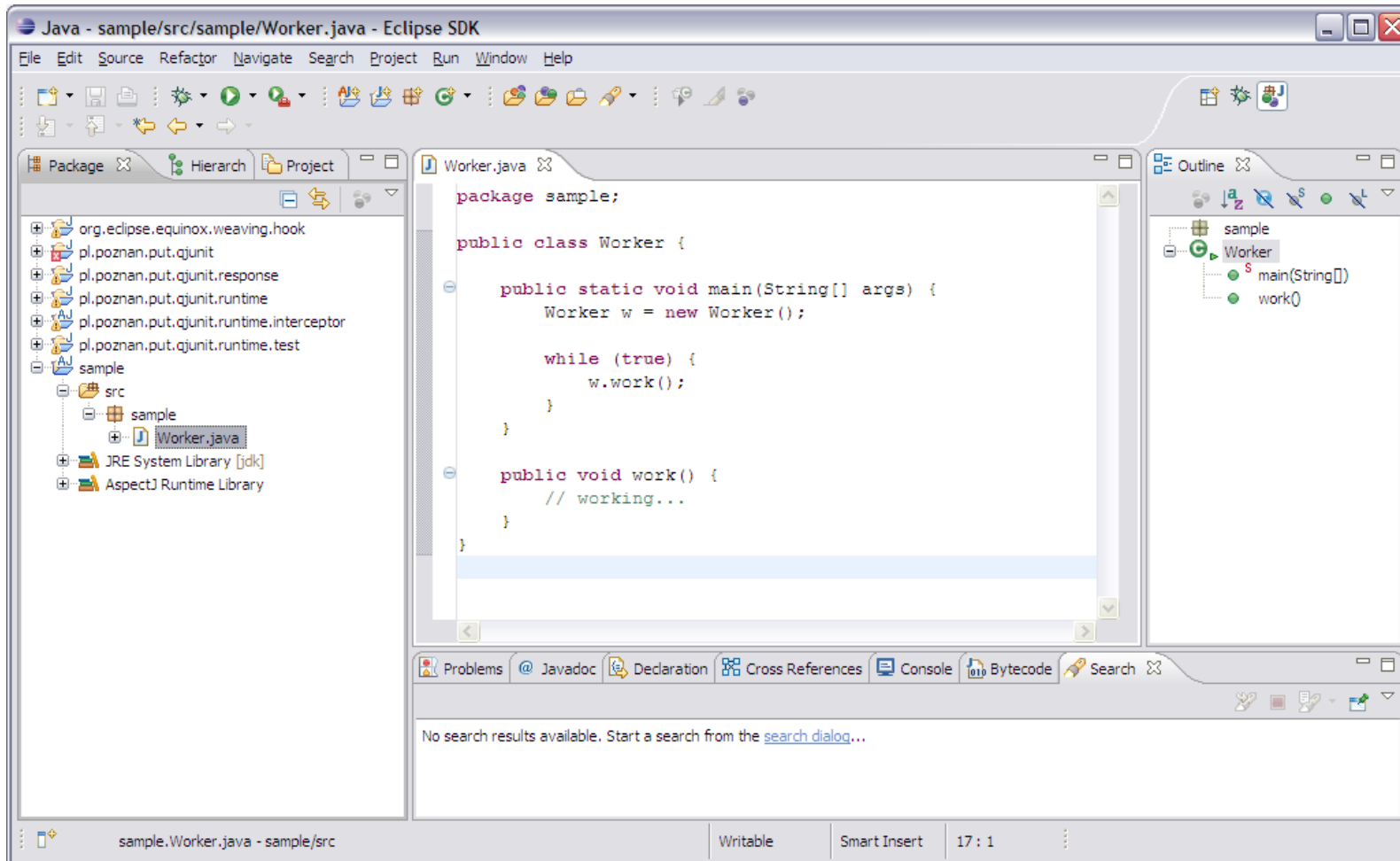
- Pojęcia
  - **Aspekt** to opis miejsc przecięcia kodu (Point-cut) oraz treści jaka ma być wpleciona pomiędzy kod (Advice)
  - **Point-cut** – definicja miejsca przecięcia kodu
  - **Advice** – kod do wplecenia
  - **JoinPoint** – konkretne miejsca przecięć (point-cutów)

# Szybki start z AspectJ (przykład)

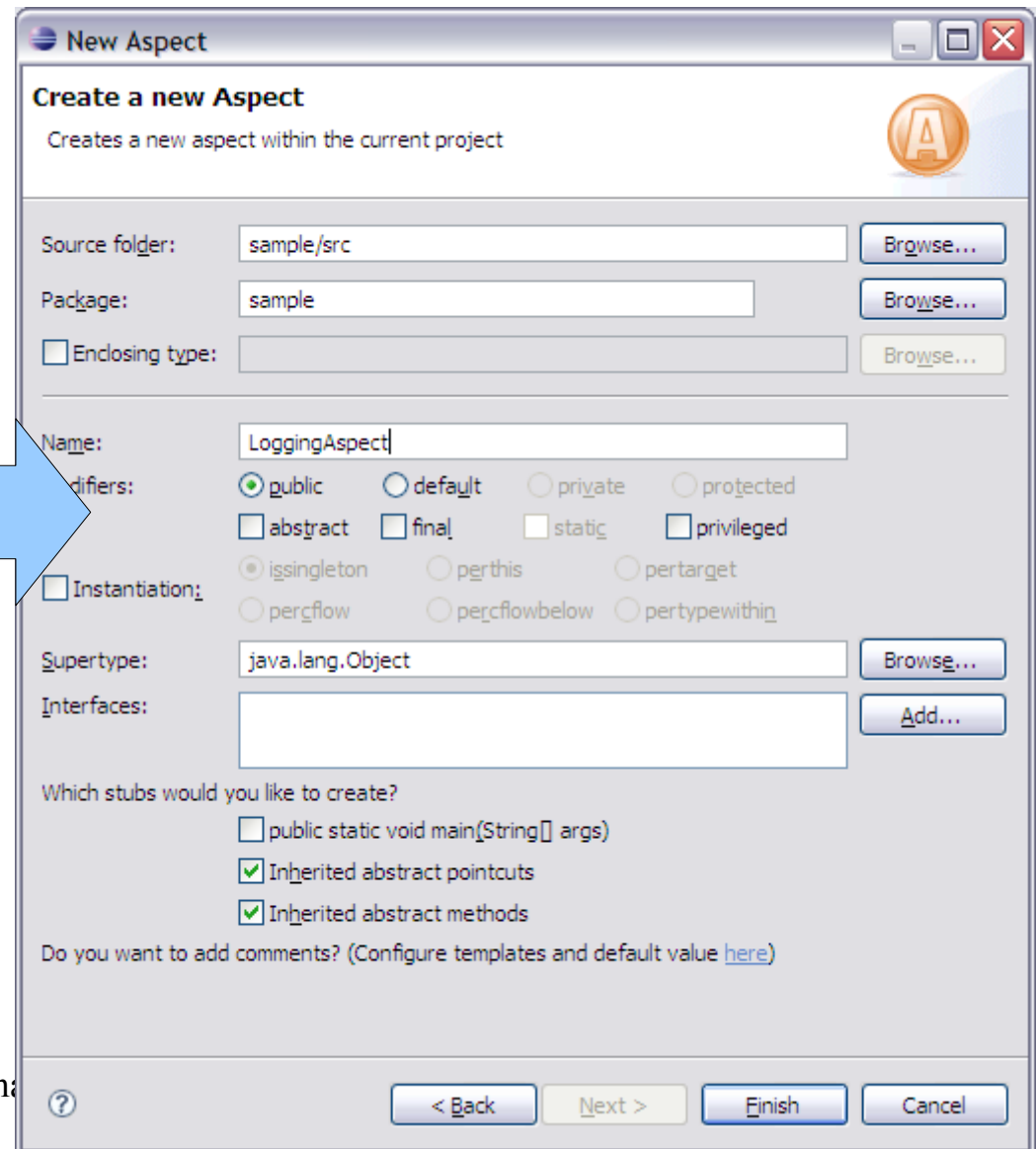
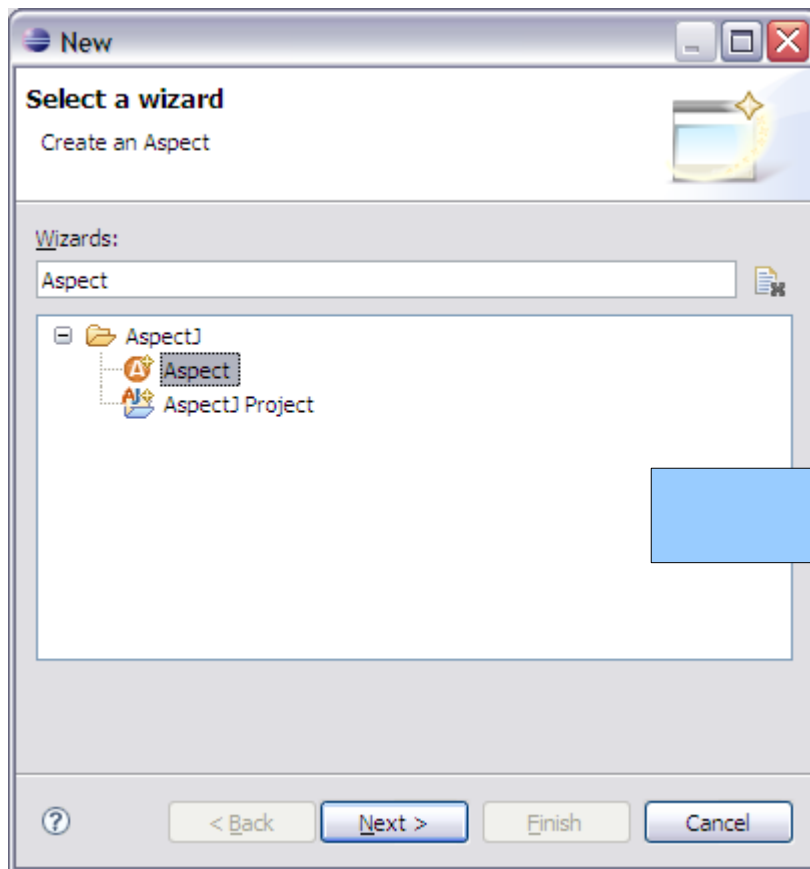
- AJDT = AspectJ Development Tools
  - Plug-in Eclipse
  - Update site:  
<http://download.eclipse.org/tools/ajdt/34/update>



# Przykładowy projekt



# Tworzemy aspekt



# Nasz pierwszy aspekt

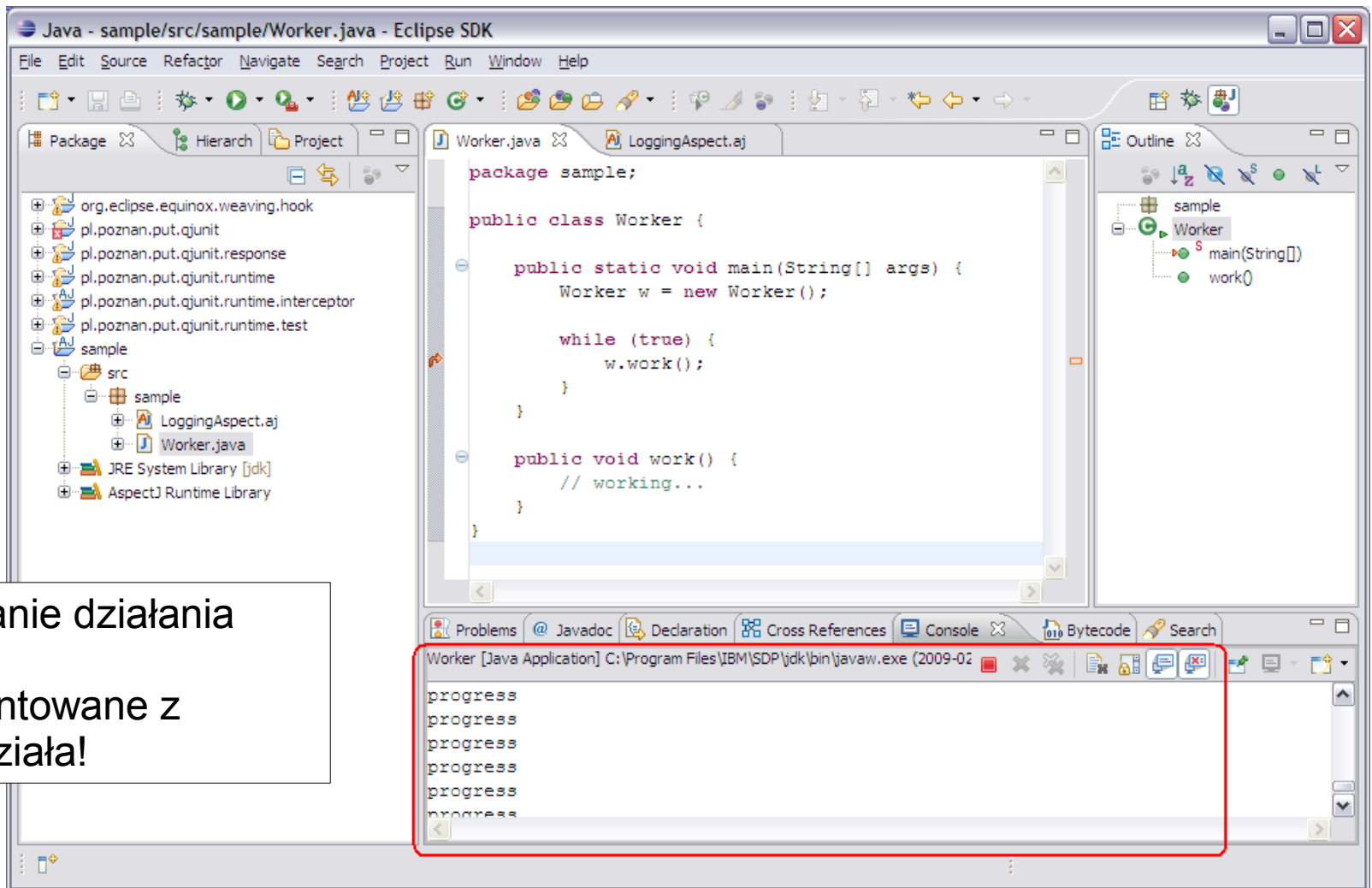
```
package sample;

public aspect LoggingAspect {

    pointcut logProgress() : call(* Worker.work(..));

    before(): logProgress() {
        System.out.println("progress");
    }
}
```

# Program z utworzonym aspektem



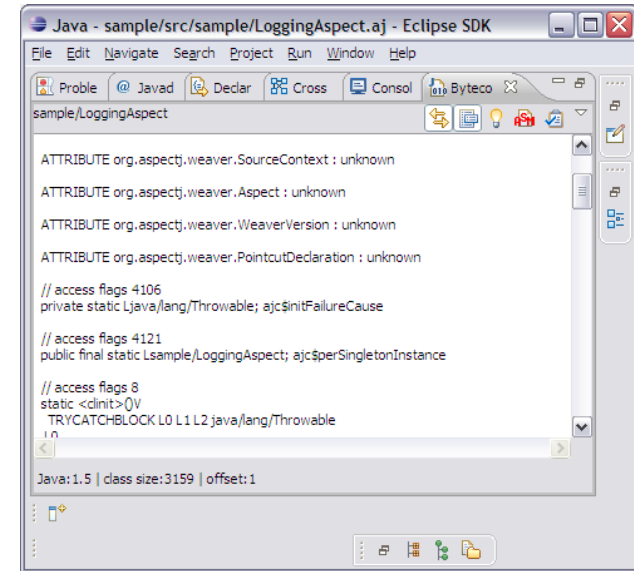
Monitorowanie działania programu zaimplementowane z aspekcie działa!

# Uruchamianie aplikacji z aspektami

- Kompilacja kodu z aspektami
  - kompilator ajc
  - Podczas uruchamiania wymagane aspectjrt.jar
- Uruchamianie i wplatanie aspektów w trakcie działania programu
  - Projekt equinox-aspects
  - Rozszerzenie Equinox OSGi

# Jak to działa

- Kompilacja kodu z aspektami
  - Kompilacja do Javy
  - Przegląd generowanego bytecode



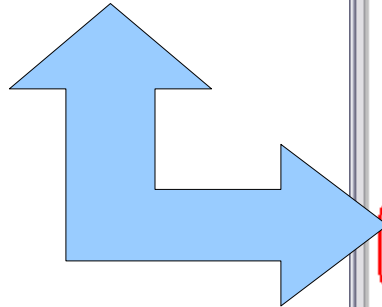
```
ATTRIBUTE org.aspectj.weaver.SourceContext : unknown
ATTRIBUTE org.aspectj.weaver.Aspect : unknown
ATTRIBUTE org.aspectj.weaver.WeaverVersion : unknown
ATTRIBUTE org.aspectj.weaver.PointcutDeclaration : unknown
// access flags 4106
private static Ljava/lang/Throwable; ajc$initFailureCause
// access flags 4121
public final static Lsample/LoggingAspect; ajc$perSingletonInstance
// access flags 8
static <clinit>()V
TRYCATCHBLOCK L0 L1 L2 java/lang/Throwable
in
Java: 1.5 | class size: 3159 | offset: 1
```

- Wplatanie konspektów w trakcie działania programu (Equinox-Aspects)
  - Rozszerzenia mechanizmu ładowania klas (ClassLoaderDelegateHook)

# Jak to działa: call

```
pointcut logProgress() : call(* Worker.work(..));
```

```
before() : logProgress() {  
System.out.println("progress");  
}  
}
```

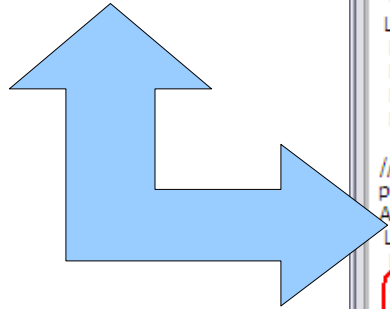


```
Java - sample/src/sample/Worker.java - Eclipse SDK  
File Edit Navigate Search Project Run Window Help  
Problems Javadoc Declaration Cross References Console Bytecode Search  
sample/Worker  
MAXSTACK = 1  
MAXLOCALS = 1  
// access flags 9  
public static main([Ljava/lang/String;)V  
ATTRIBUTE org.aspectj.weaver.MethodDeclarationLineNumber : unknown  
L0  
LINENUMBER 6 L0  
NEW sample/Worker  
DUP  
INVOKESPECIAL sample/Worker.<init>()V  
ASTORE 1  
L1  
LINENUMBER 9 L1  
ALOAD 1  
INVOKESTATIC sample/LoggingAspect.aspectOf()Lsample/LoggingAspect;  
INVOKEVIRTUAL sample/LoggingAspect.ajc$before$sample_LoggingAspect$1$56502a52()V  
INVOKEVIRTUAL sample/Worker.work()V  
L2  
LINENUMBER 8 L2  
GOTO L1  
L3  
LOCALVARIABLE args [Ljava/lang/String; L0 L3 0  
LOCALVARIABLE w Lsample/Worker; L1 L3 1  
MAXSTACK = 2  
MAXLOCALS = 2  
Java: 1.5 | class size: 1088
```

# Jak to działa: call

```
pointcut logProgress() : execution(* Worker.work(..));
```

```
before() : logProgress() {  
System.out.println("progress");  
}
```



```
Java - sample/src/sample/Worker.java - Eclipse SDK  
File Edit Navigate Search Project Run Window Help  
Problems Javadoc Declaration Cross References Console Bytecode Search  
sample/Worker  
L1  
LINENUMBER 9 L1  
ALOAD 1  
INVOKEVIRTUAL sample/Worker.work()V  
L2  
LINENUMBER 8 L2  
GOTO L1  
L3  
LOCALVARIABLE args [Ljava/lang/String; L0 L3 0  
LOCALVARIABLE w Lsample/Worker; L1 L3 1  
MAXSTACK = 2  
MAXLOCALS = 2  
  
// access flags 1  
public work()V  
ATTRIBUTE org.aspectj.weaver.MethodDeclarationLineNumber : unknown  
L0  
LINENUMBER 15 L0  
INVOKESTATIC sample/LoggingAspect.aspectOf()Lsample/LoggingAspect;  
INVOKEVIRTUAL sample/LoggingAspect.ajc$before$$sample_LoggingAspect$1$56502a52()V  
L1  
RETURN  
LOCALVARIABLE this Lsample/Worker; L1 L1 0  
MAXSTACK = 1  
MAXLOCALS = 1  
}  
  
Java: 1.5 | class size: 1088
```

# Możliwości wplatania kodu

- Wywołanie metody (call/execution)
- Odwołanie/zmiana wartości pola (get/set)
- Tworzenie obiektu  
(call/execution/initialization/preinitialization)
- Tworzenie klasy (staticinitialization)
- Obsługa wyjątku (handler)
- Obsługa advice (adviceexecution)
- ...inne

# Możliwości wplatania kodu

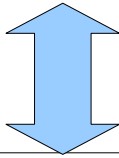
- Przed / zamiast / po / po rzuceniu wyjątku / po nie rzuceniu wyjątku
  - before
  - around
  - after
  - after throwing
  - after returning

# Przykład

- Środowisko do testowania mutacyjnego
- Przecina bibliotekę JUnit, by zebrać dodatkowe dane na temat uruchamiania testów

# Wykorzystane aspekty

```
pointcut systemCallBoolean() : this(junit.framework.TestCase)  
&& withincode(* test*(..)) && call(boolean *(..));
```



Miejsce przecięcia w każdej klasie typu `junit.framework.TestCase` (czyli także klasy dziedziczące), wewnątrz metod o nazwie zaczynającej się na „test”, o dowolnych argumentach („..”), zwracających dowolną wartość („\*”) w każdym odwołaniu („call”) do dowolnej metody zwracającej wartość typu `boolean`.

# Odpowiedzi na pytania

- *Testowanie /miluch*
  - *czyli czy jest możliwość bez żadnej wtyczki sprawdzenia czy w danym punkcie dany advice zostanie uruchomiony...*
  - *testowanie advice*
- *Używanie AspectJ do wprowadzenia policy dotyczących architektury systemu: np sprawdzanie pewnych metryk w kodzie, zarządzanie wyjątkami. /miluch*

# Dziękuję za uwagę

## •Referencje

- AspectJ: [www.eclipse.org/aspectj](http://www.eclipse.org/aspectj)
- AJDT: <http://www.eclipse.org/ajdt>
- AspectJ Programming Guide:  
[www.eclipse.org/aspectj/doc/released/progguide](http://www.eclipse.org/aspectj/doc/released/progguide)