Practical Webobjects
Chapter 1:

Making Your Code Better

WR

WR@Csus4.net

http://www.csus4.net/WR/

目次

Eclipse and WOLips/WOProject

- What Is Eclipse?
- What Is WOLips/WOProject?
- Why Use Eclipse and WOLips/WOProject?
- Issues with Eclipse
- Pros and Cons

Why Use Eclipse and WOLips/WOProject?

1. EclipseとWOLips/WOProjectを使いなさい。
   - ご利益は（習得の）努力に値する。
   - まぁ、乗り換えなくても、この本からたくさんのことを得ることができるけどね。
   - 使うことをオススメするけど、mustじゃあない。
2. Windows(とMacの併用)ならば使いなさい。
   - Windows環境はAppleから見捨てられている。《やっぱし?》
   - PB on MacとPBWO on Windowsを併用することは悪夢だ。
     - 《XCodeは?》
   - Eclipseなら併用OK！

Issues with Eclipse

1. 問題点がいくつかあるけど、イイ！
   - 問題点
     - 安定性
     - OS Xで遅い
     - アプリケーション起動、ユニットテストの際にCLASSPATH絡みの設定が必要
   - でもイイ！
     - Eclipse とWOLips/WOProjectの組み合わせは最高！
     - ご利益は（習得の）努力に値する（くどい！）

Pros and Cons (1)

1. Apple純正開発環境と比較してみるよ。
2. Pros: 長所
   - Fully functionalなWindows開発(?)
   - クロス・プラットフォーム開発
   - 多数のオープンソースプロジェクト
     - Eclipse本体とプラグイン
   - クロス・プラットフォーム開発！（くどい！）
   - 生産性を高めるツール (or 仕組み)
     - テンプレート、パースペクティブ、コード補完
   - クロス・プラットフォーム開発について触れたっけ？
     （いや、マジでくどいから！）

Pros and Cons (2)

3. Cons: **短所**
   - OS XではUI絡みの挙動がのろい。
   - Javaソースファイルの配置をpackageと同じくする必要あり！
     - 既存プロジェクトの移行が面倒
     - 特にソースコード管理ツールを使っている場合は！
   - リソース管理がイマイチ
     - リソースファイルの手動コピー
     - ビルドスクリプトの編集
   - クラスパス設定が必要となる場合あり
   - 若干の安定性の問題
4. (略)

Automated Unit Testing

- The Case for Automated Unit Testing
- JUnit: Java Unit Testing
- JUnit and EOF: DbUnit

The Case for Automated Unit Testing

1. 単体テスト(Unit Testing)を自動化せよ。
2. 自動単体テストはドキュメントになりうる。
   - クラスの使用例として機能する。
   - しかも、原理的に「コードとドキュメントの乖離」が起こりえない。
3. 自動単体テストはコード(の機能性・品質)を保証する。
4. Caution:
   1. 良いテストコードを書くには鍛錬が必要。
   2. 外部テストチーム (or品質保証チーム) が実施するブラックボックステストに代わるものではない。

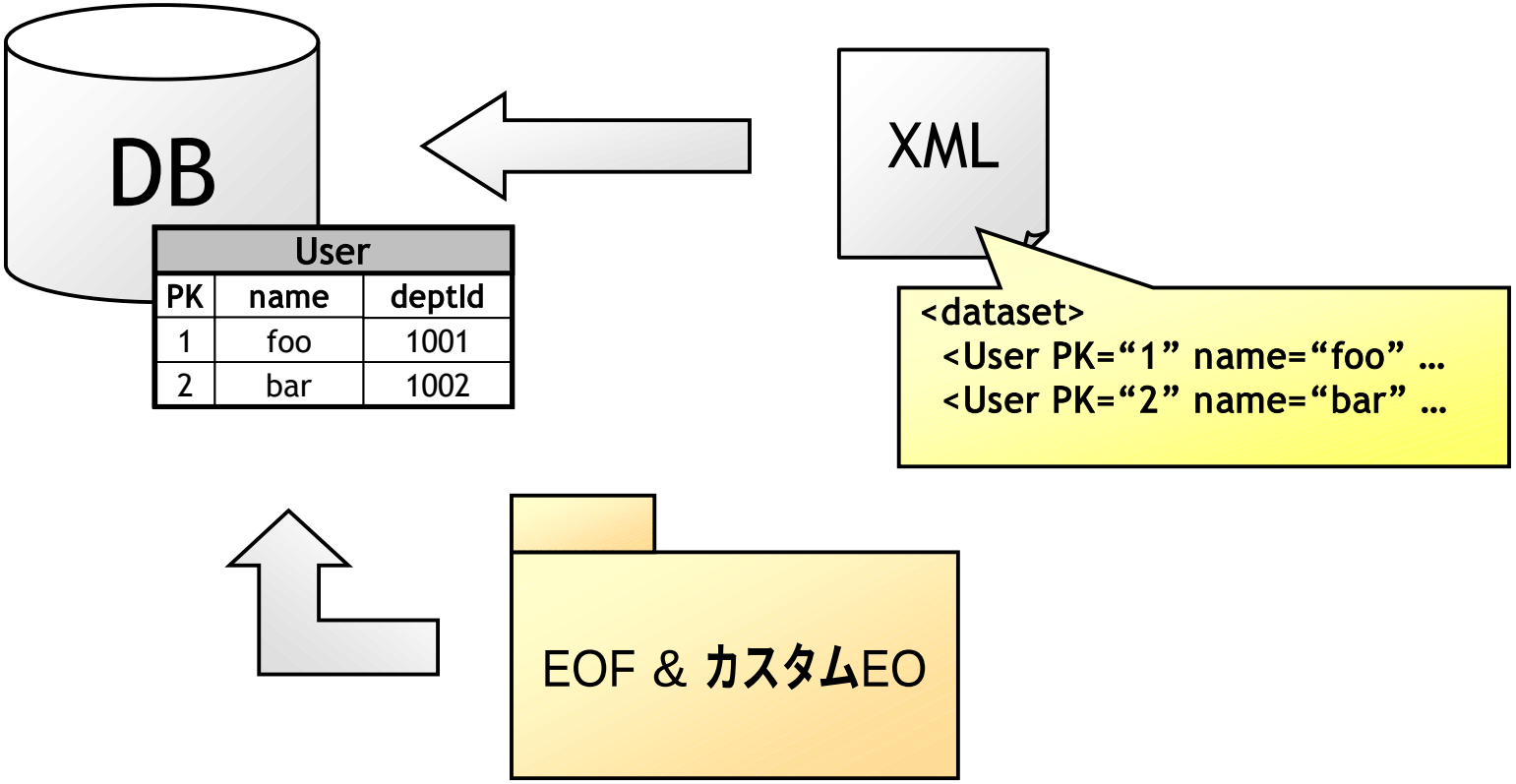
JUnit: Java Unit Testing

- (略)
- いのっちさんのプレゼンをキチンと聴きましょう!!!

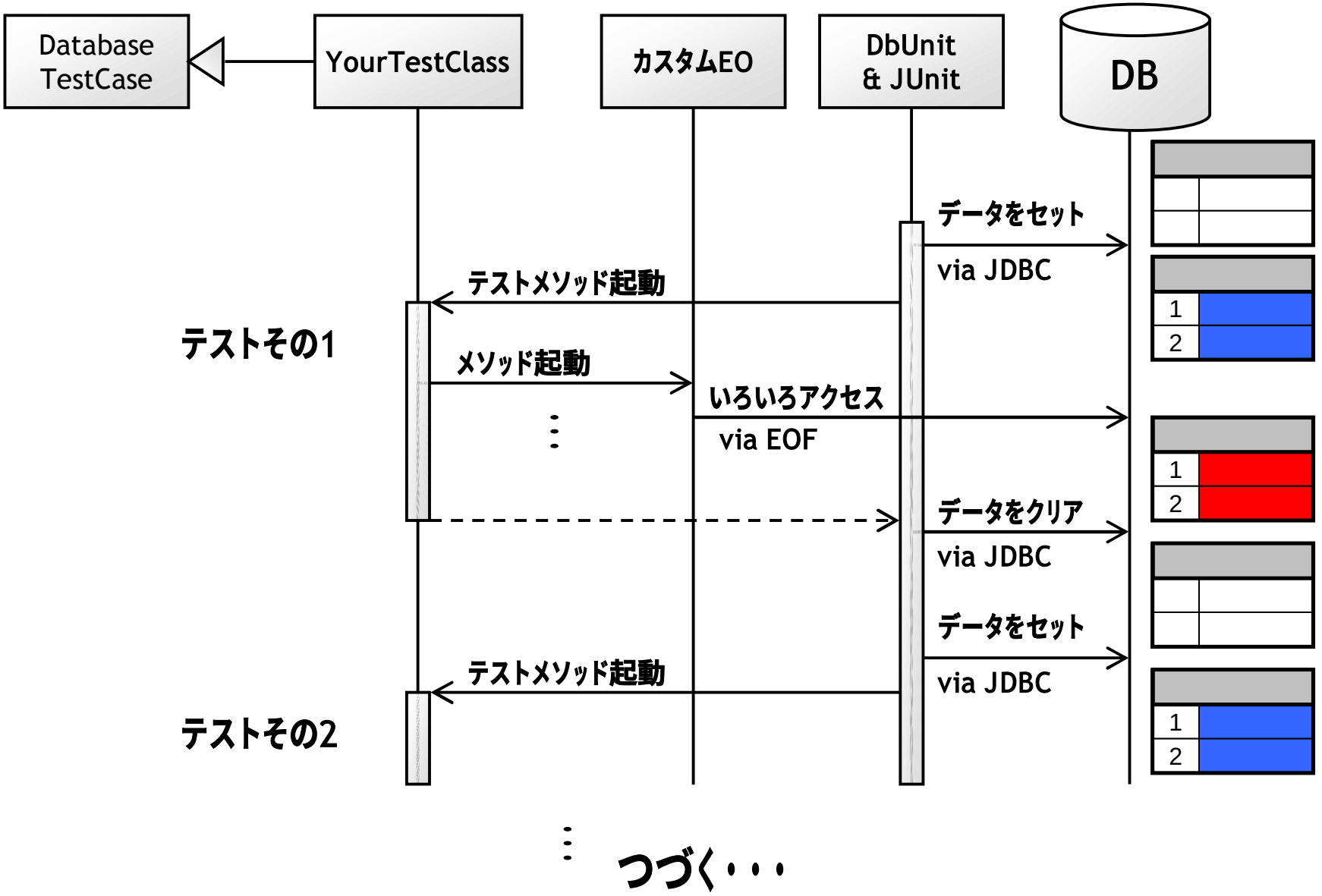
JUnit and EOF: DbUnit

1. DBUnitはEOの自動単体テストに役立つ
   - WebObjectsの鍵はEOFである。
   - DBUnitはDBのデータをセットアップしてくれるJUnitの拡張。
2. DBUnitはWOAをテストする際の問題を解決する。
   - WOAをテストするためには、DBにデータをセットアップする必要がある。
   - DBUnitはこの問題を解決する。

DbUnit Concepts

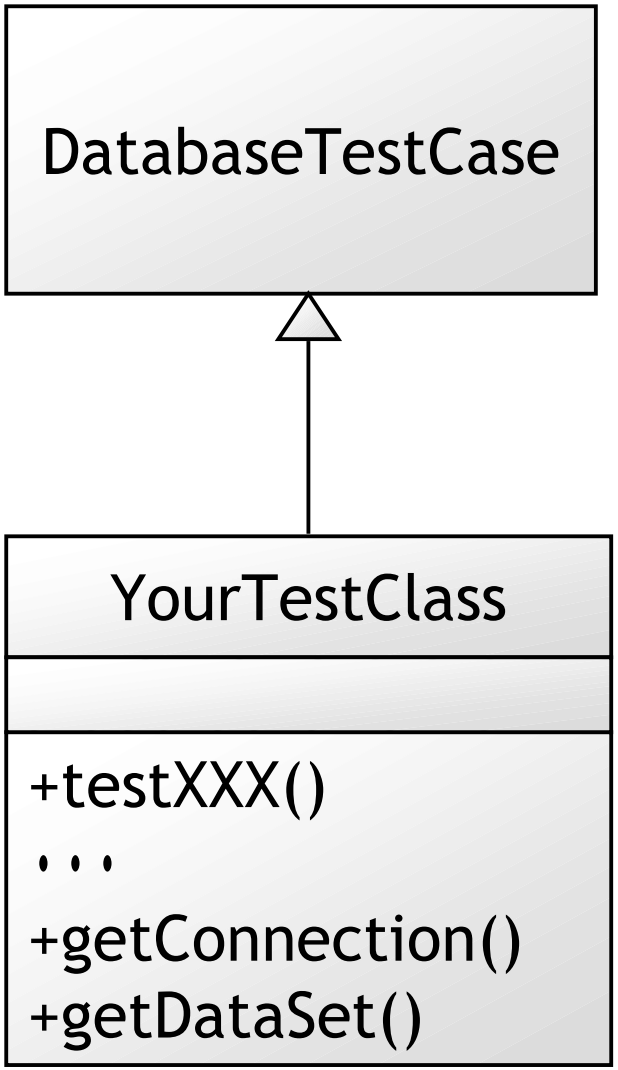


DbUnit Sequence



Using DbUnit

1. DbUnitの使い方
   - 基本的にJUnitと同じ
     - testXXX()を書けばよい。
   - ただし、DatabaseTestCaseクラスを継承して、getConnection()とgetDataSet()をオーバライドせよ
   - すると、勝手にDBのセットアップと後始末をやってくれる
2. テストコード（セットアップと後始末）はDBアクセスにEOFを使わない
   - 普通にJDBCを使う



Using DbUnit (cond.)

3. カスタマイズと拡張
   - 基本的には、カスタマイズ/拡張の必要はない。
   - セットアップと後始末をカスタマイズ/拡張したい場合は、スーパークラスの作成、設定項目のファイル化をしたくなるかも。
   - 我々は、DatabaseTestCaseのサブクラスを作って、以下を実現している
     - ecのセットアップ
     - 設定ファイルからのコネクション文字列読み出し
     - ユーティティメソッドの提供

Design by Contract

- What Are Contracts?
- DBC and Unit Testing
- What Design by Contract Is Not
- More on Design by Contract
- Why Contract Your Code?
- Java and DBC

- **WebObjectsに関係ない一般論に終始しており、あえて語る部分もないので、省略**

Debugging Techniques : Logging in WebObjects

- Using Subclasses
- Delegates
- Using Notifications
- NSLog and Other Logging Tools
- Snooping on EOF
- Digging Deeper: Seeing the Fetched Data
- The Last Log Entry

- **構成イマイチ・・・なので、変えてみる。**

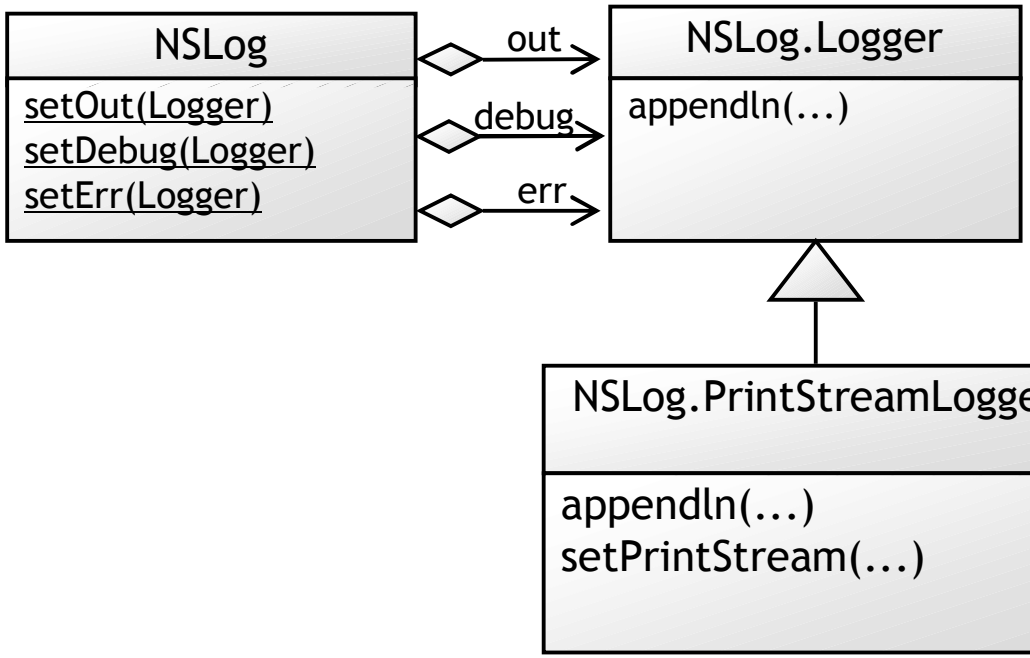
Debugging Techniques：Logging in WebObjects

- **ロギング用ライブラリ/クラス**
  - NSLog and Other Logging Tools
    - **標準**：NSLog
    - 3rd Party：Other Logging Tools
- **うまくロギングを実現するTips**
  - Using Subclasses
  - Delegates
  - Using Notifications
- **ロギング事例**
  - Snooping on EOF
  - Digging Deeper: Seeing the Fetched Data
- **説教**
  - The Last Log Entry

NSLogとLog4J

	NSLog	Log4J
ロギング対象	フレームワーク or ユーザコード	**基本的にユーザコード**
ログ出力レベル	DebugLevel	Priority
ログ出力フィルタ	**定義済み** DebugGroup (ユーザー定義も可能)	**ユーザ定義**category (クラス名と同じくするのが普通)
出力先	java.io.PrintStream	Appender
レイアウト	ー	Layout

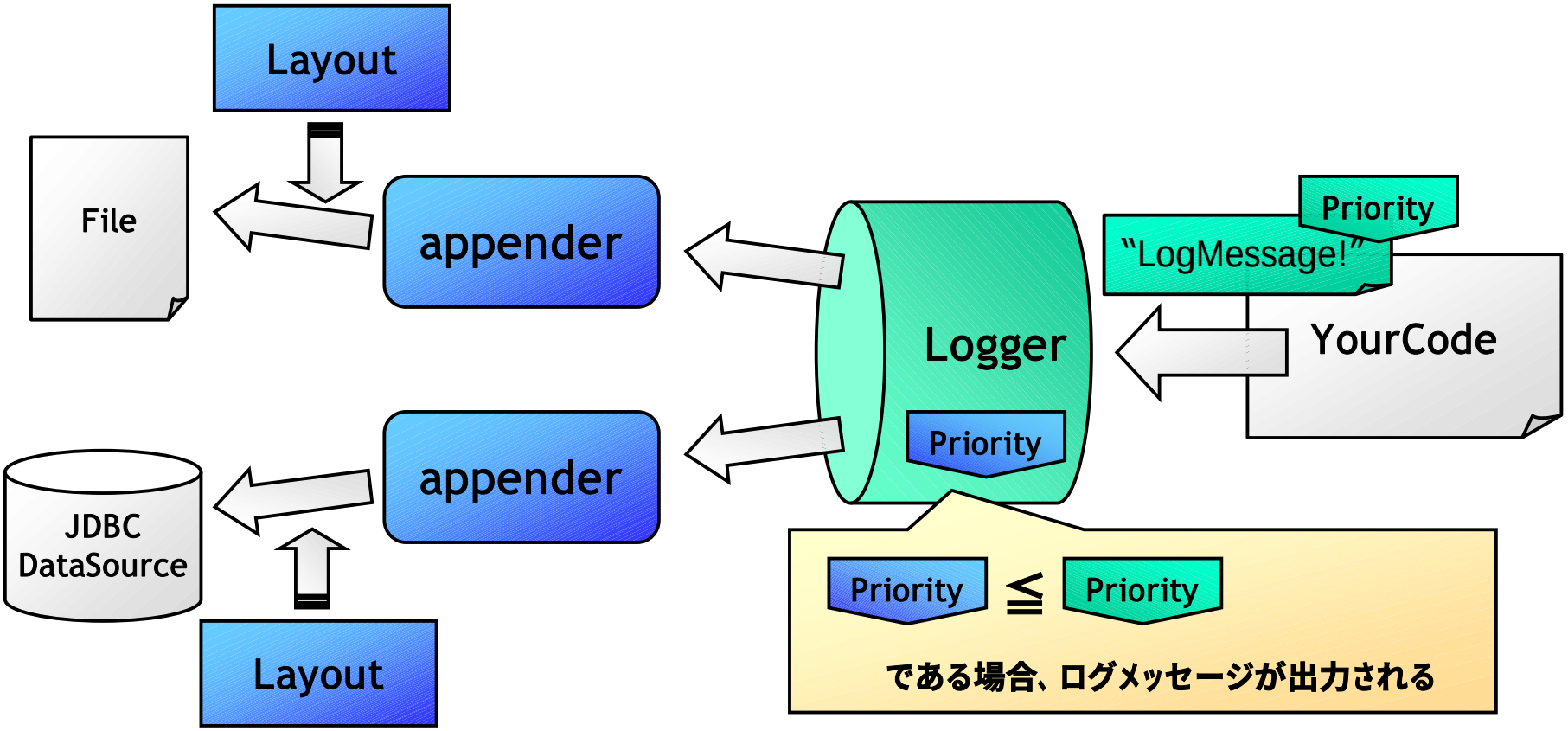
NSLog Class Diagram




```
[Usage]
NSLog.out.appendln("info");
NSLog.debug.appendln("debug");
NSLog.err.appendln("err");
```

NSLog	WebObjectsのロギングシステムにアクセスするためのスタティッククラス (って呼び方あったっけ?)
NSLog.Logger	ログ出力先に対応した抽象クラス。 NSLog.Loggerのサブクラスを NSLogのout, debug, errに登録する。
NSLog.PrintStreamLogger	ログをjava.io.PrintStreamに出力するロガー。 NSLog.Loggerの具象サブクラス

Log4J Concepts



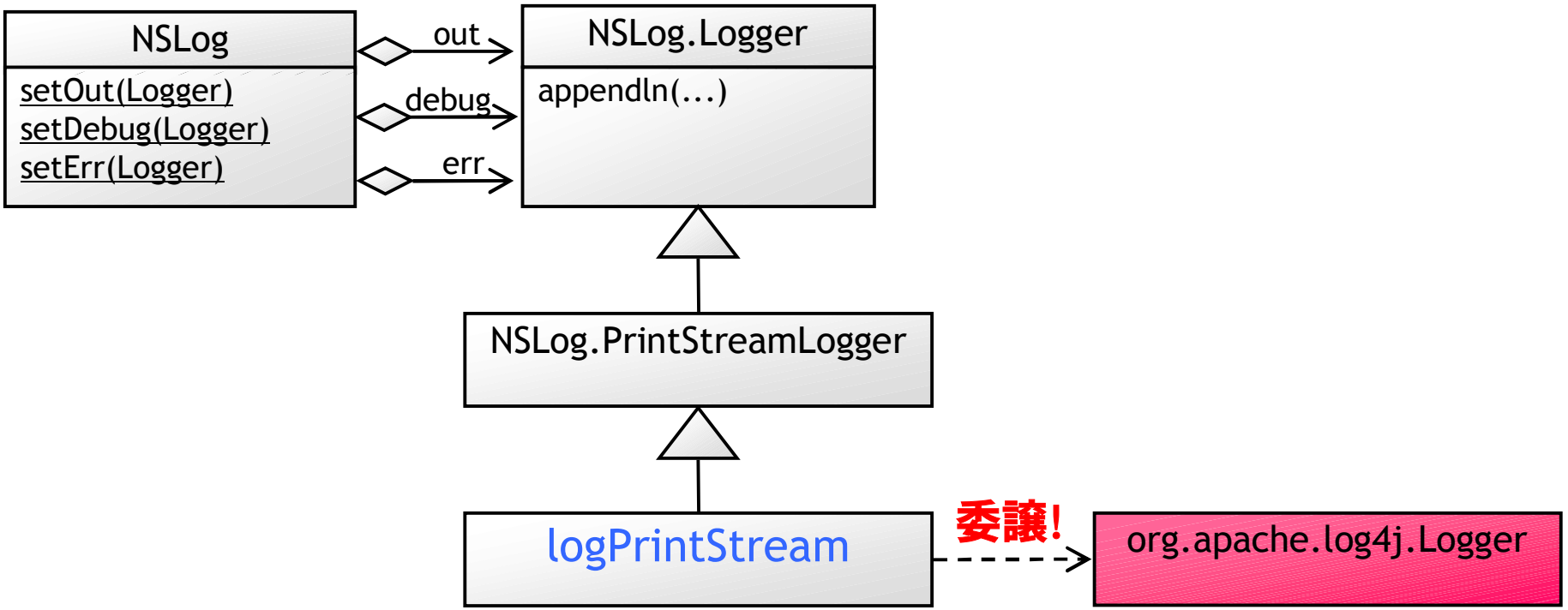
予めコードに埋め込んでおくもの

(一般に) 設定ファイルで組み上げるモノ

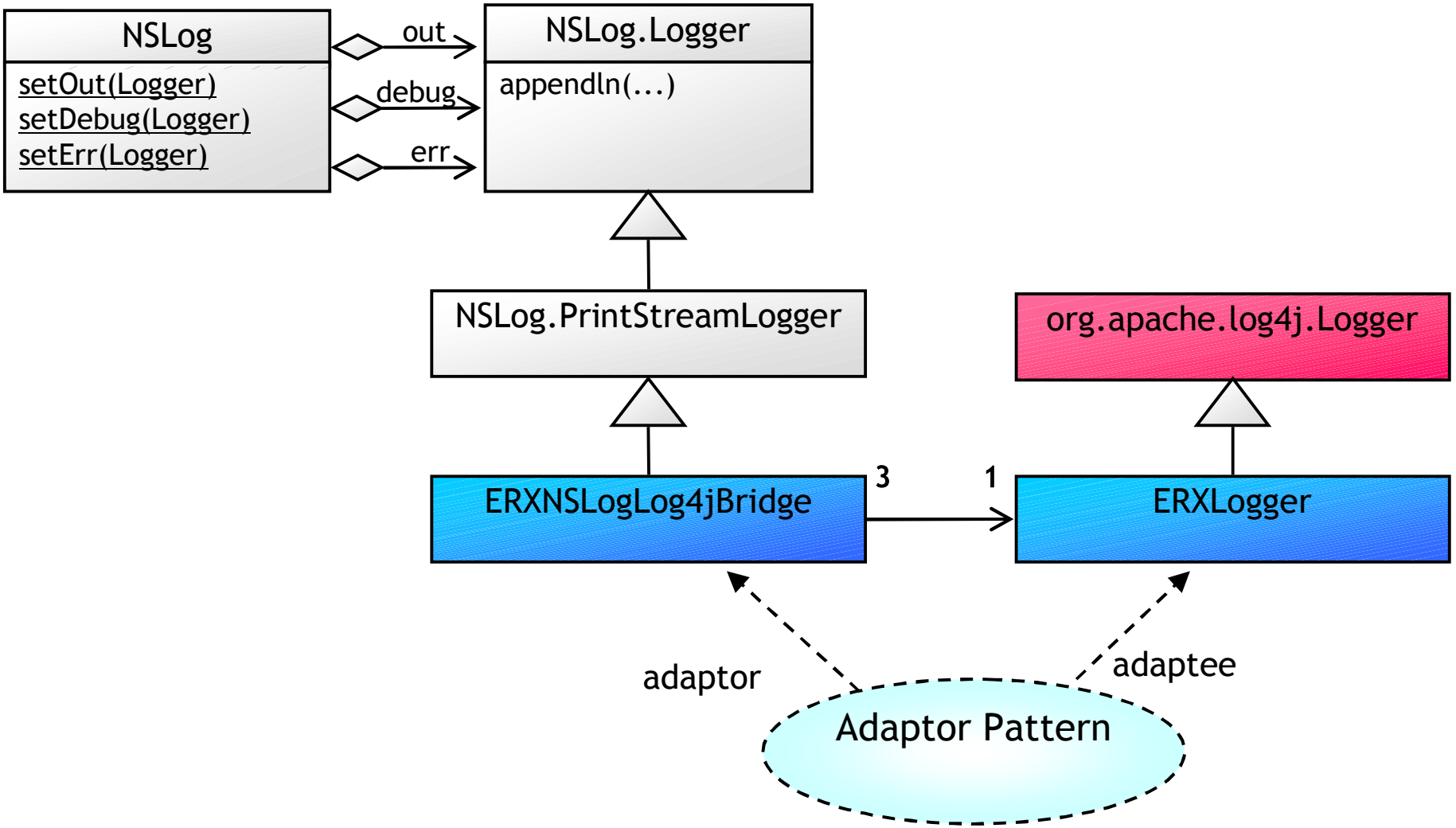
Loggerの階層構造については省略...

NSLog → Log4J (page 23)

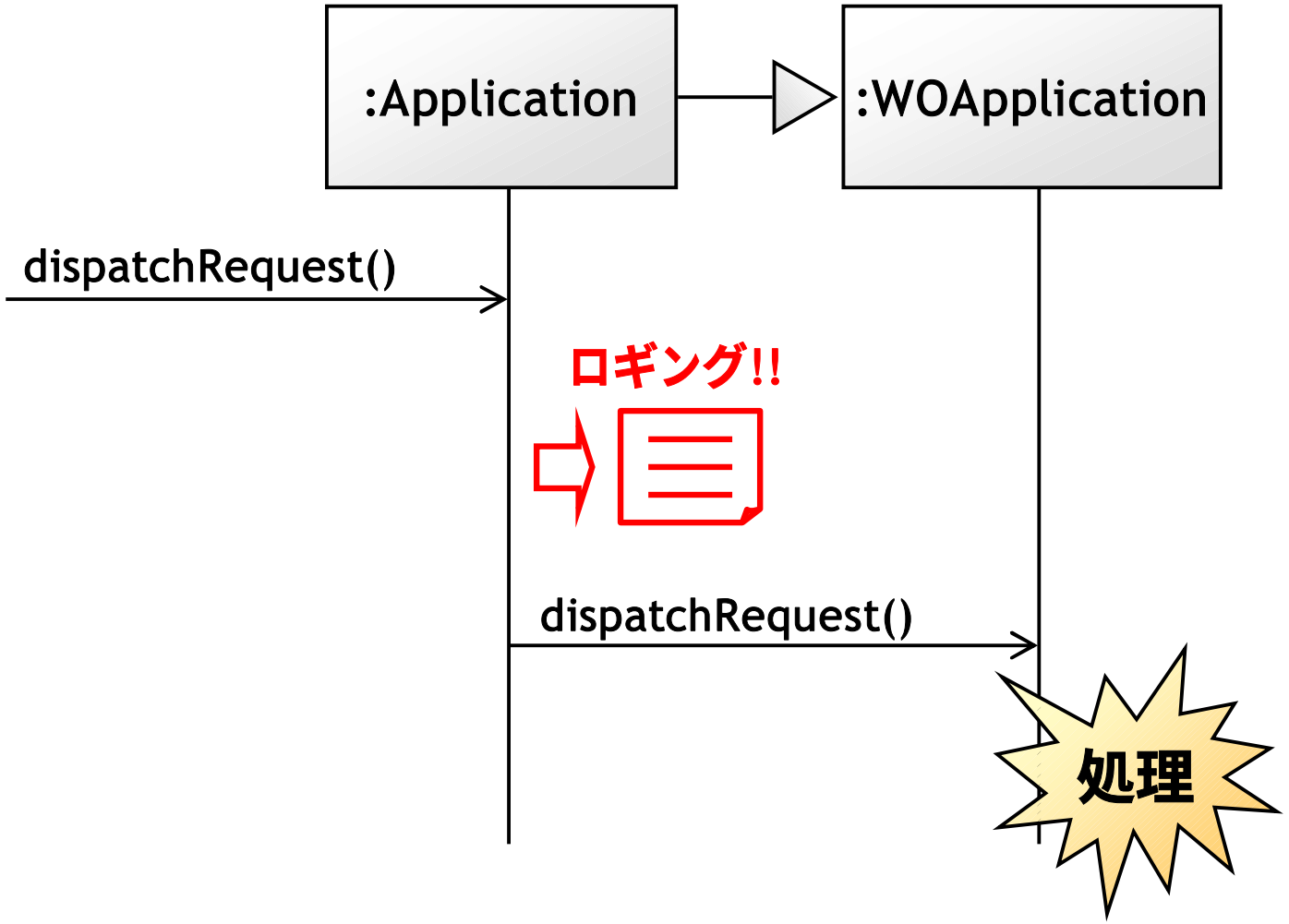
```
((NSLog.PrintStreamLogger)NSLog.out).setPrintStream(logPrintStream)
((NSLog.PrintStreamLogger)NSLog.debug).setPrintStream(logPrintStream)
((NSLog.PrintStreamLogger)NSLog.err).setPrintStream(logPrintStream)
```



おまけ：ERXNSLogLog4jBridge



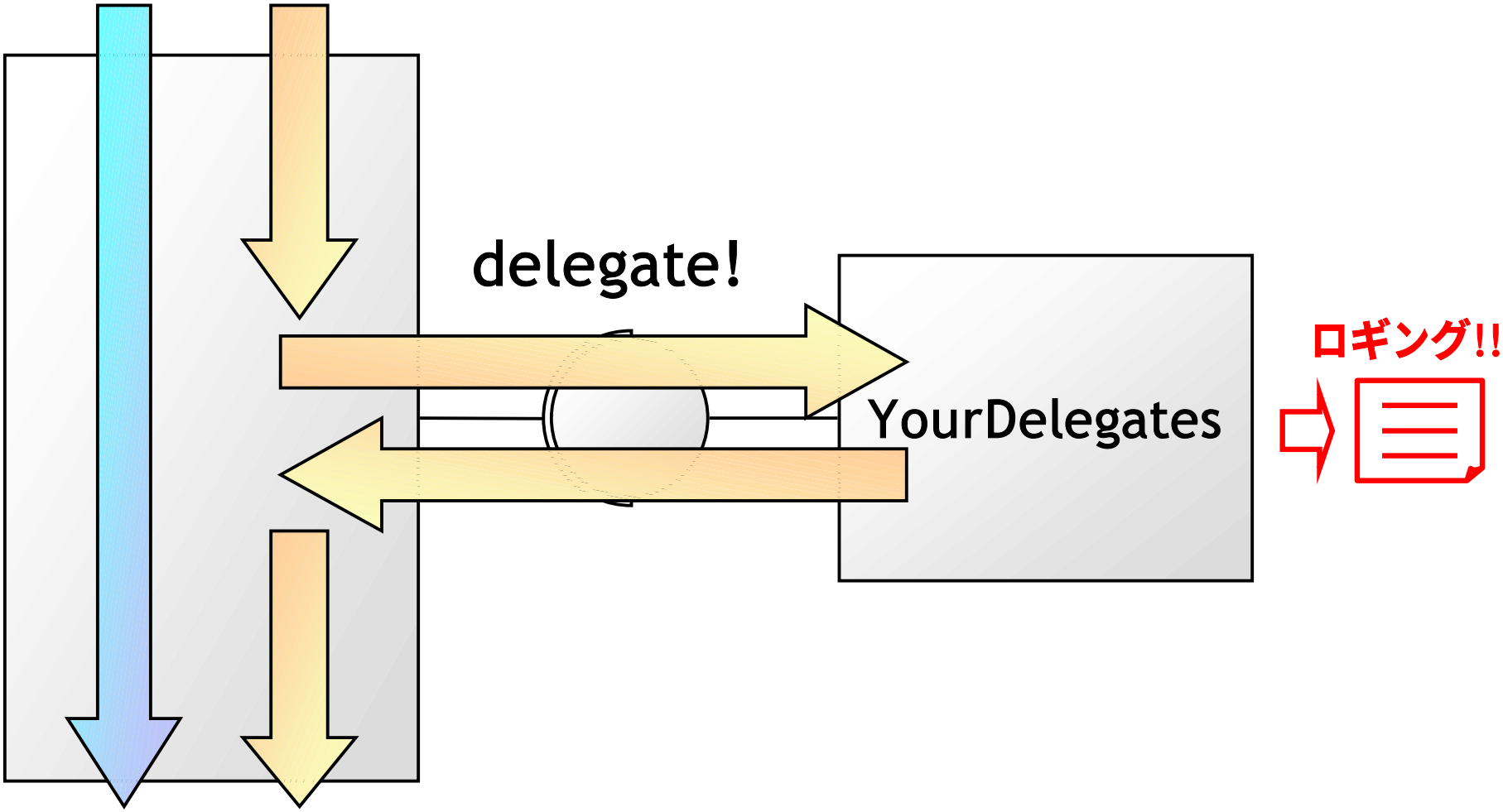
Using Subclasses



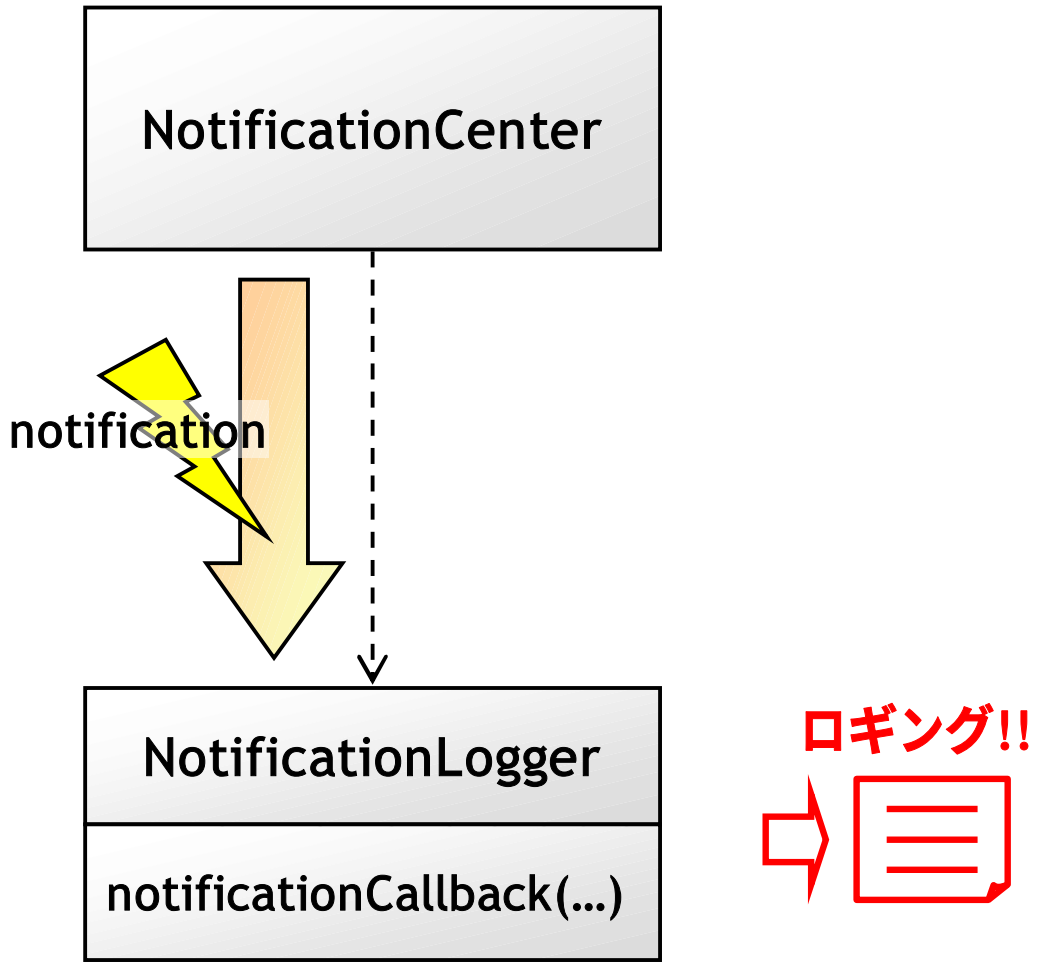
継承を用いた親クラスへの委譲パターンという見方もできる。

まあ、WOF/EOF自体が継承ベースのフレームワークだからなぁ・・・

Delegates Concept & Logging



Using Notifications



Important Notification

Class	Notification
EOClassDescription	ClassDescriptionNeededForClassNotification
	ClassDescriptionNeededForEntityNameNotification
EOEditingContext	ObjectsChangedInEditingContextNotification
EOAdaptorContext	AdaptorContextBeginTransaction
	AdaptorContextCommitTransaction
	AdaptorContextRollbackTransaction
EOModelGroup	ModelAddedNotification

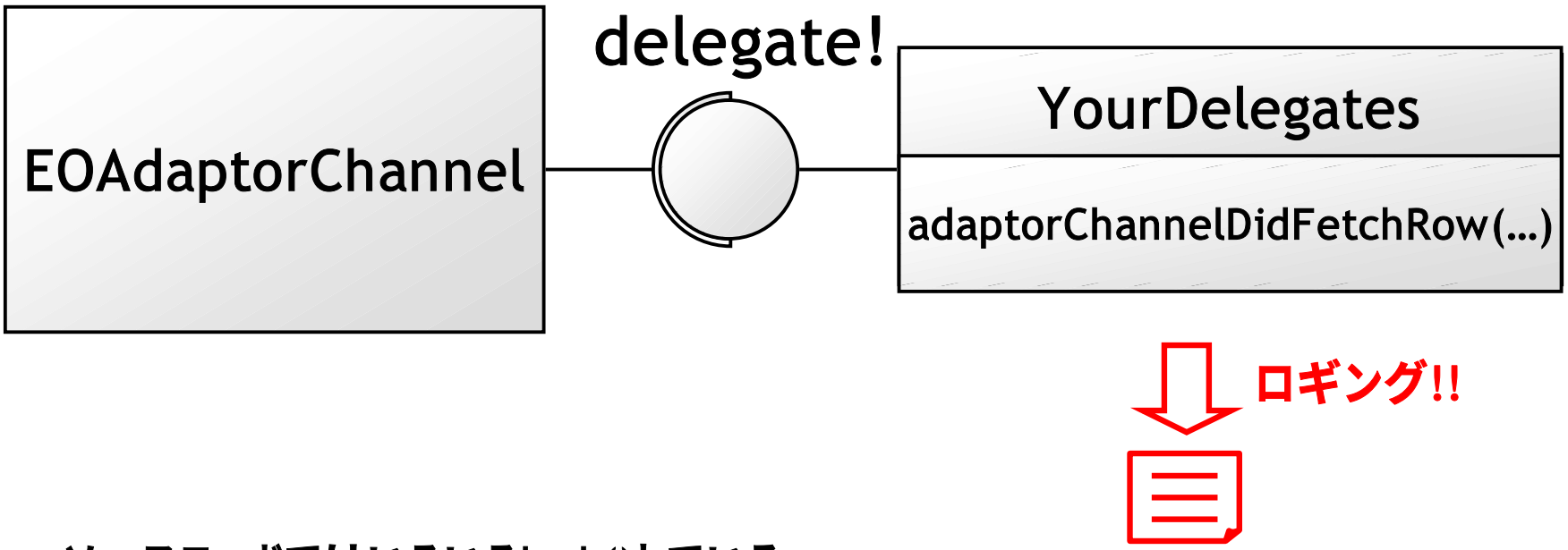
Snooping on EOF

- -DEOAdaptorDebugEnabled=true
  - いつもログ出力されて邪魔
  - → DebugGroupを動的に変更するが吉

```
public static void logSQL(boolean shouldLog){
    if (shouldLog){
      NSLog.allowDebugLoggingForGroups(NSLog.DebugGroupSQLGeneration
                                       | NSLog.DebugGroupDatabaseAccess
                                       | NSLog.DebugGroupEnterpriseObjects);
   }else{
     NSLog.refuseDebugLoggingForGroups(NSLog.DebugGroupSQLGeneration
                                       | NSLog.DebugGroupDatabaseAccess
                                       | NSLog.DebugGroupEnterpriseObjects);
   }
}
```

Digging Deeper: Seeing the Fetched Data

- フェッチしたデータを見たい。
- しかし、フェッチしたデータを見せてくれるような DebugGroupはフレームワーク側で定義されていない。
  → Delegateでフレームワークをカスタマイズする。



ソースコードではいろいろlock()している
ようですが、よくわかりません。

The Last Log Entry

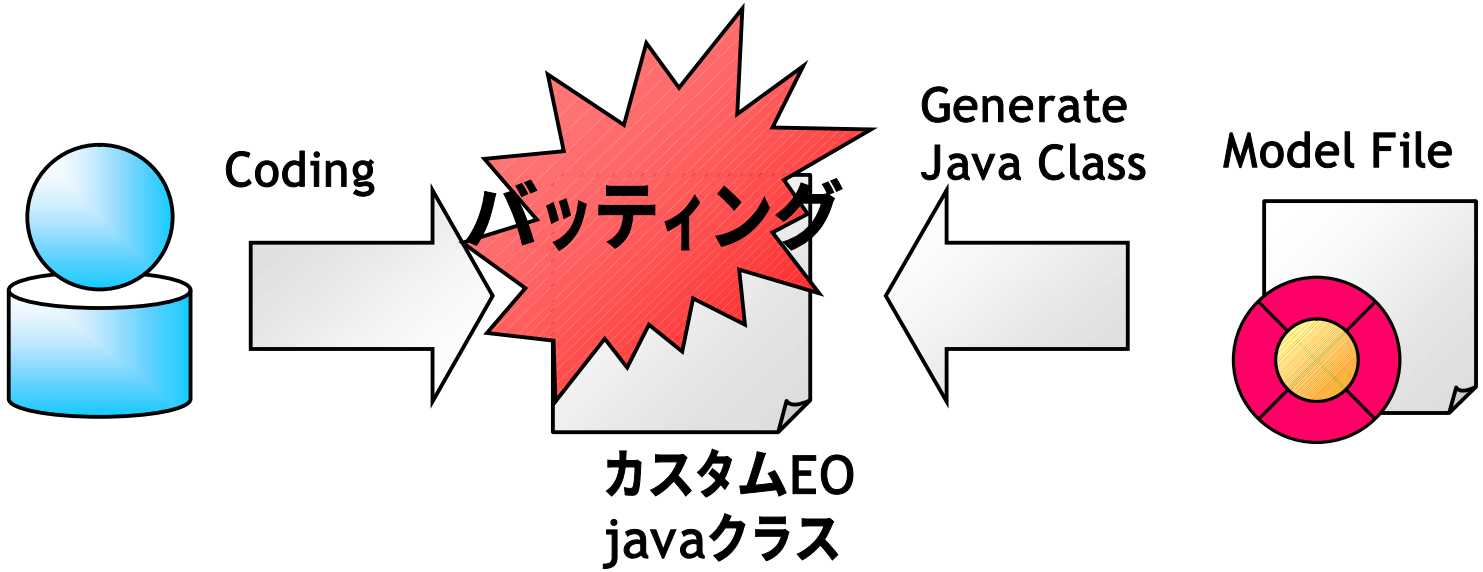
- デバッガより、Loggingを使いなさい。

Creating Java Classes from EOModels

- The Generation Gap Pattern
- The Trecty EOGenerator
- Alternatives to EOGenerator

The Generation Gap Pattern

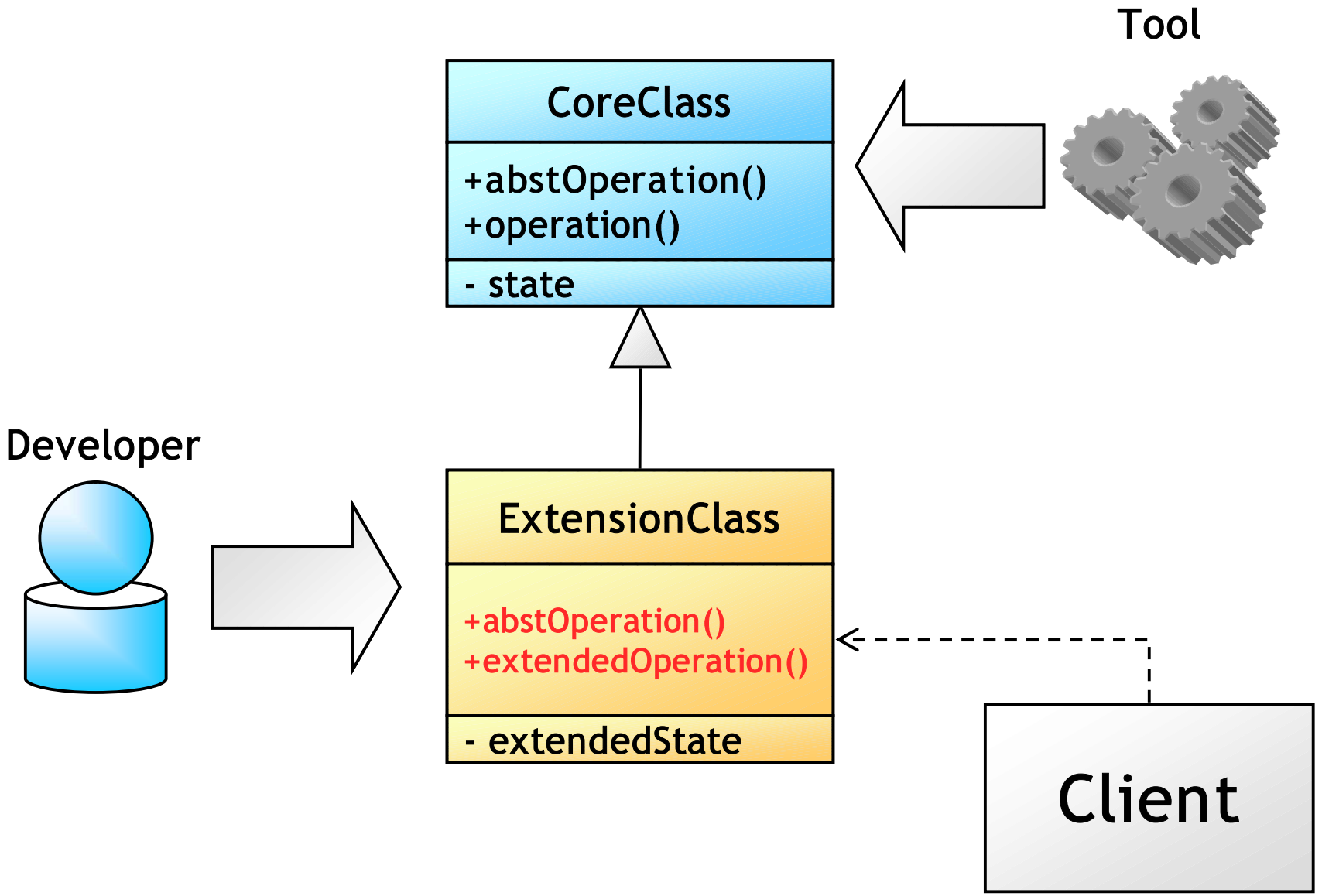
- Problem
  - ツールが自動生成したコードと、デベロッパがコーディングしたコードがバッティングする
  - WebObjects開発においては…



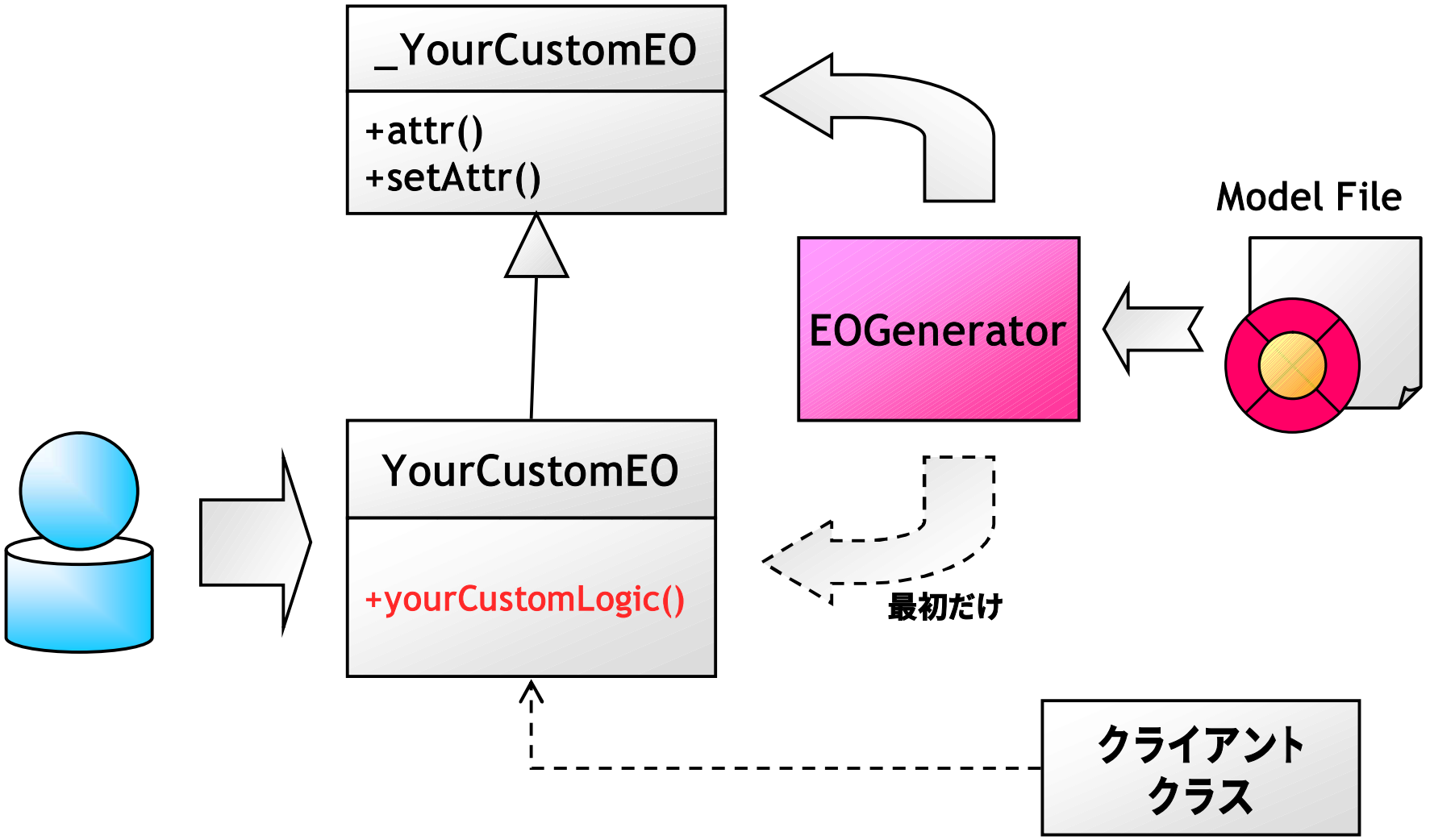
→ マージ、まじうざい。　　(ふぅ・・・)

→ つーわけで、Generation Gap パターン

Generation Gap Pattern Concept



EOGenerator



おしまい

- 次回はどうしましょうか???