

# yui-frameworks



AKABANA



yui-frameworks

AKABANA 有川榮一

# 自己紹介

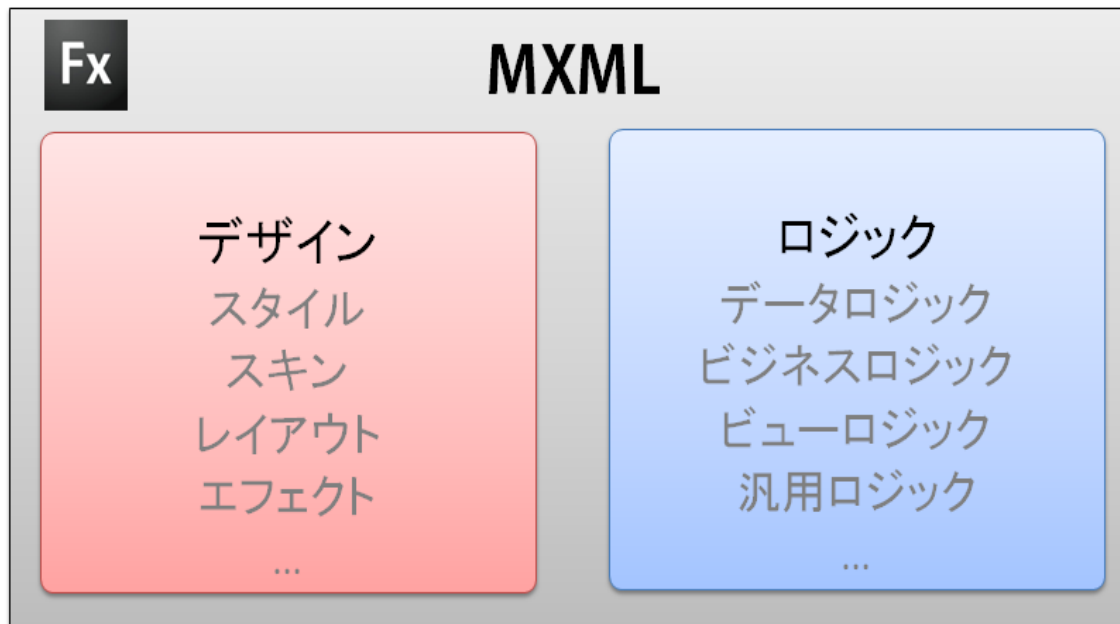
- 有川 榮一
  - AKABANA
    - RIA, Mobile, Cloud, Training
    - <http://www.akabana.net>
  - The Seasar Project コミット
    - S2Flex2
    - AKABANA / yui-frameworks
  - Blog
    - <http://akabana.info>

# アジェンダ

- yui-frameworks
  - アーキテクチャ
  - 開発ルール

- 特徴
  - 国産フレームワーク
  - 軽量
  - Flex3, Flex4, Flex4+Catalyst対応

- アーキテクチャ
  - デザインとロジックの分離



# yui-frameworks

## ■ アーキテクチャ

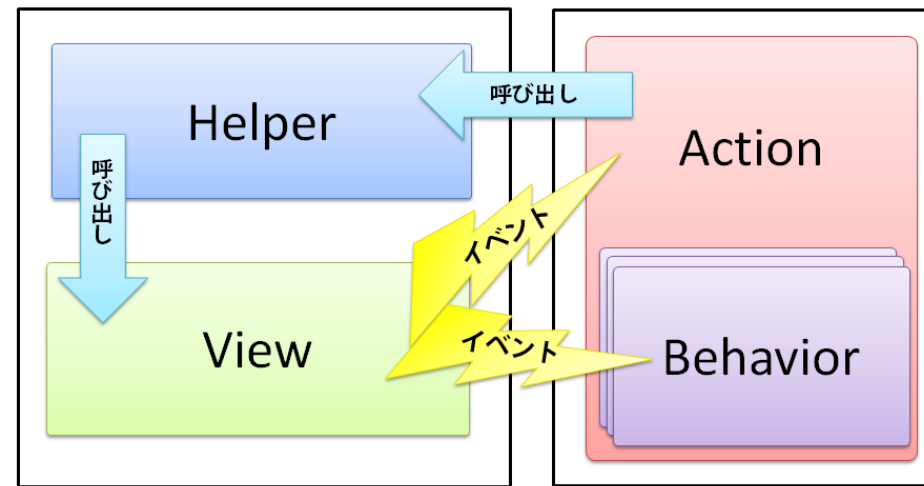
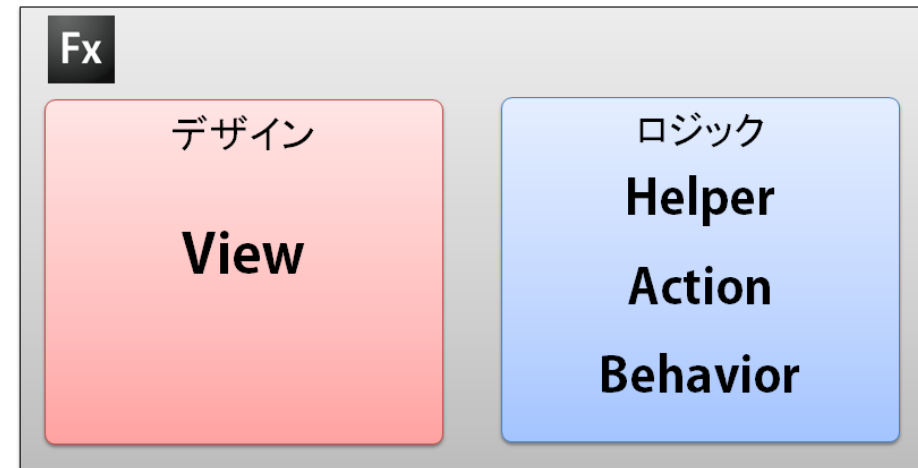
### ■ 4つのクラス役割

- View
- Helper
- Action
- Behavior

### ■ デザインとロジックの分離

### ■ デザインとロジックを結合

- 実行時に結合



## ■ 機能

- 自動イベントハンドリング
  - addEventListenerやremoveEventListenerを記述しなくてもいい
  - ハンドラの引数をなしにできる
- ルールベースのDI
  - クラス役割間の依存関係を解決するため
- サービス
  - RPC : S2Flex2,T2
  - DS : BlazeDS
  - Local : 通常のASクラスをサービスとみなす

# 開発ルール

# yui-frameworks 開発ルール

- 対象パッケージ指定
  - yui-frameworksの対象となるパッケージを指定
  - conventions.propertiesファイルで指定
    - package=examples.helloworld (, ... )\*

# yui-frameworks 開発ルール

- クラス役割の命名規則

- View

- {対象パッケージ}.view. {画面名}View

- Helper

- {対象パッケージ}.helper. {画面名}Helper

- Action

- {対象パッケージ}.action. {画面名}Action

- Behavior

- {対象パッケージ}.behavior. {画面名}{操作名}Behavior

# yui-frameworks 開発ルール

## ■ クラス役割の命名規則

### ■ View

- {対象パッケージ}.view. {画面名}View

### ■ Helper

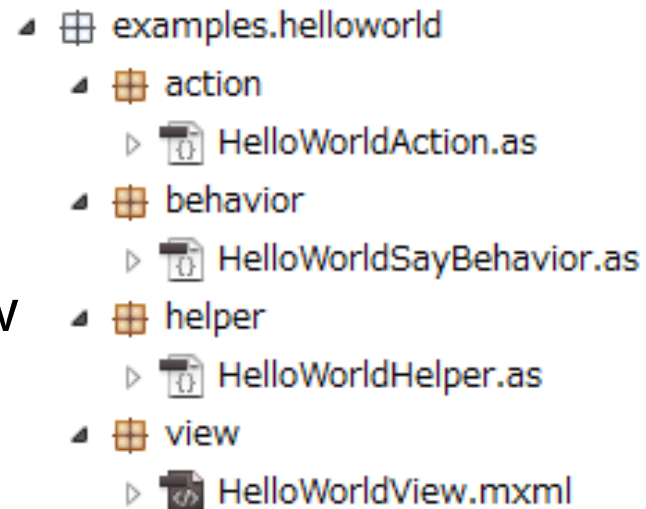
- {対象パッケージ}.helper. {画面名}Helper

### ■ Action

- {対象パッケージ}.action. {画面名}Action

### ■ Behavior

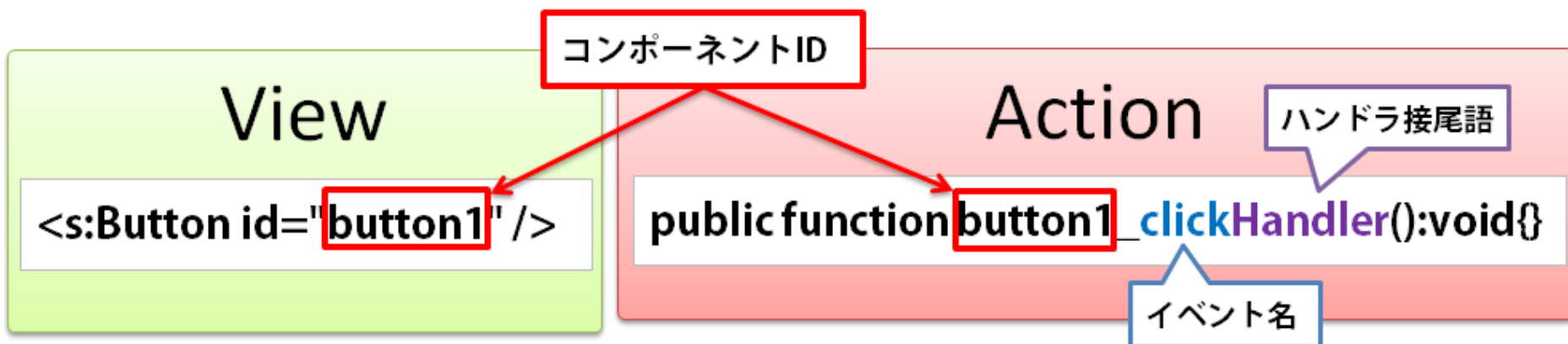
- {対象パッケージ}.behavior. {画面名}{操作名}Behavior



# yui-frameworks 開発ルール

## ■ イベントハンドラ名の命名規則

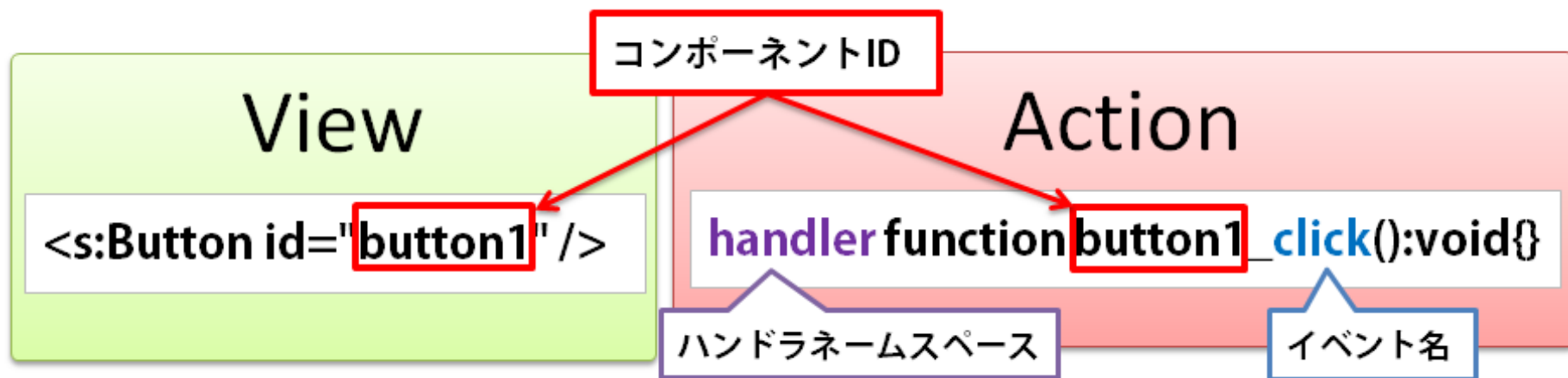
- `public {コンポーネントID}{大イベント名}Handler(event:Event):void{`
- `public {コンポーネントID}_{イベント名}Handler (event:Event):void{`



# yui-frameworks 開発ルール

## ■ イベントハンドラ名の命名規則

- handler {コンポーネントID}{大イベント名}(event:Event):void{
- handler {コンポーネントID}\_{イベント名}():void{



# yui-frameworks 開発ルール

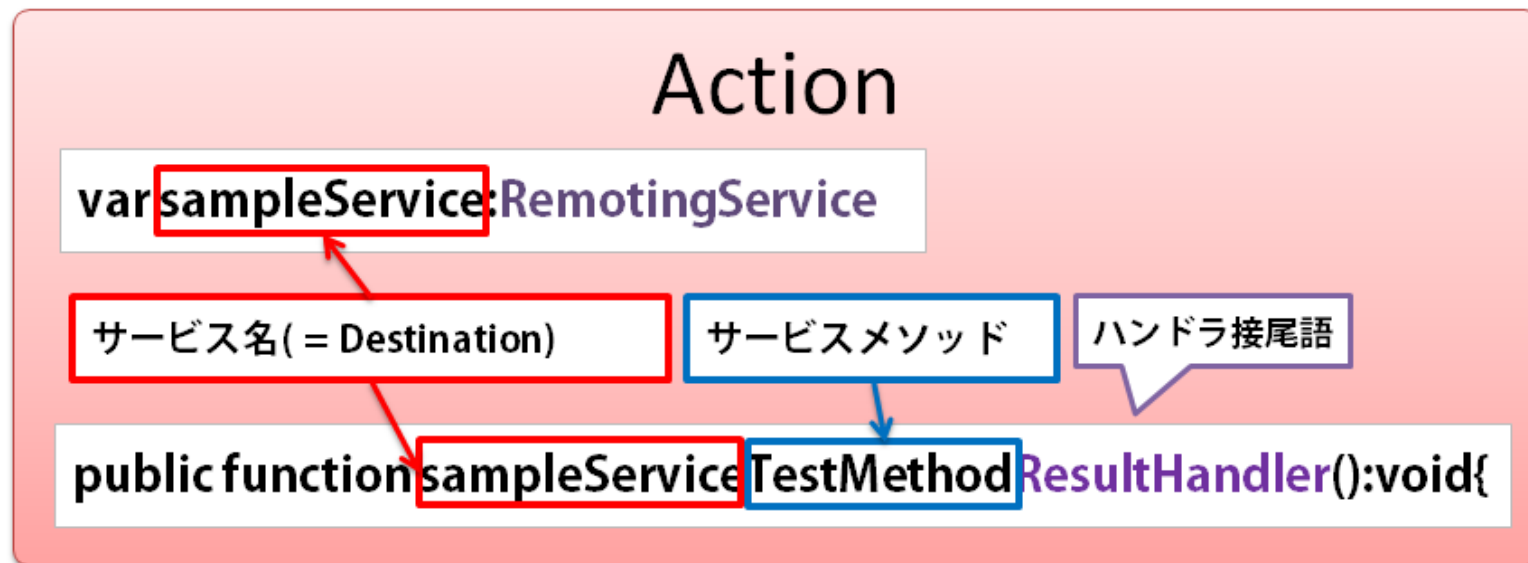
## ■ サービスレスポンスメソッド名の命名規則

### ■ 成功時

- `public {サービス変数名} {メソッド名}ResultHandler(event:ResultEvent):void{`

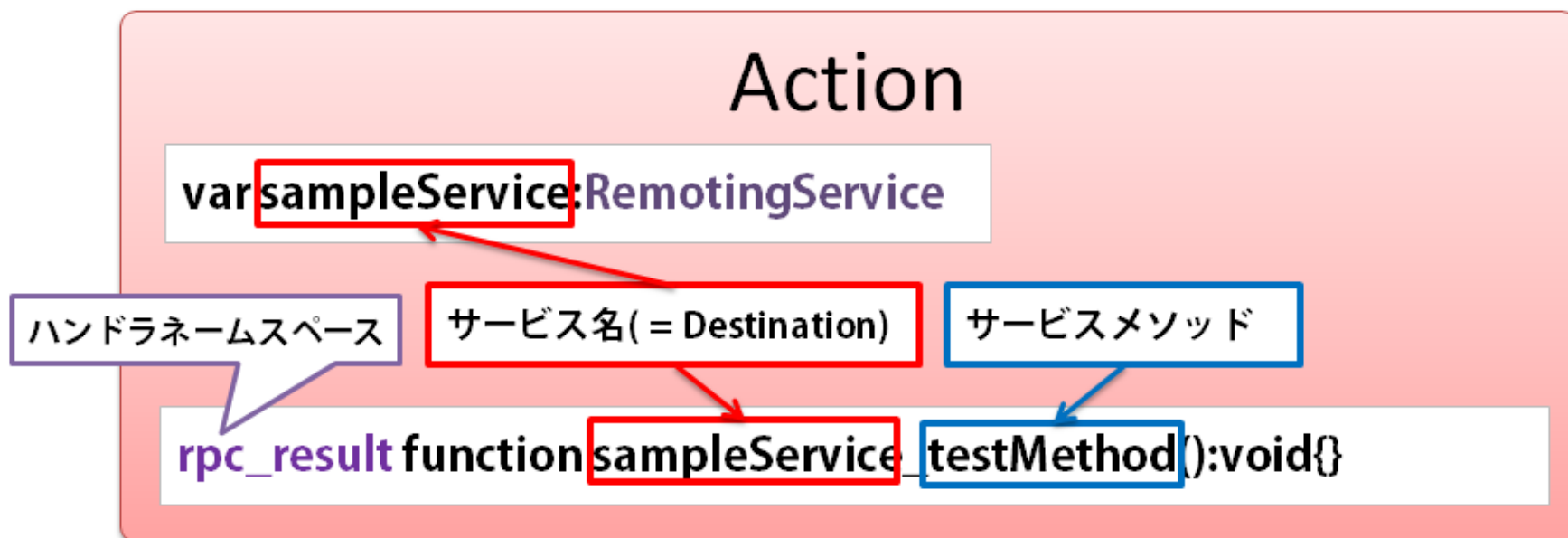
### ■ 失敗時

- `public {サービス変数名} {メソッド名}FaultHandler(event:FaultEvent):void{`



# yui-frameworks 開発ルール

- サービスレスポンスメソッド名の命名規則
  - 成功時
    - `rpc_result {サービス変数名}_{メソッド名}(event:ResultEvent):void{`
  - 失敗時
    - `rpc_fault {サービス変数名}_{メソッド名} (event:FaultEvent):void{`



**理解を深めるために**

# 参考URL

- 記事

- [yui-frameworksの必要性とアーキテクチャについて](#)

- ドキュメント

- <http://yui-docs.akabana.info/>

- トレーニング

- <http://www.akabana.net>

ご清聴  
ありがとうございました