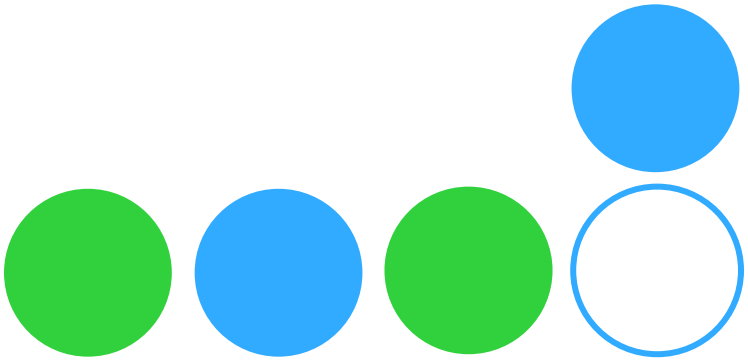
Webコンテナ活用法再考

- SDLoaderでServletとFlexとSWTと

片山　暁雄
株式会社キャピタルアセットプランニング
T2プロジェクト

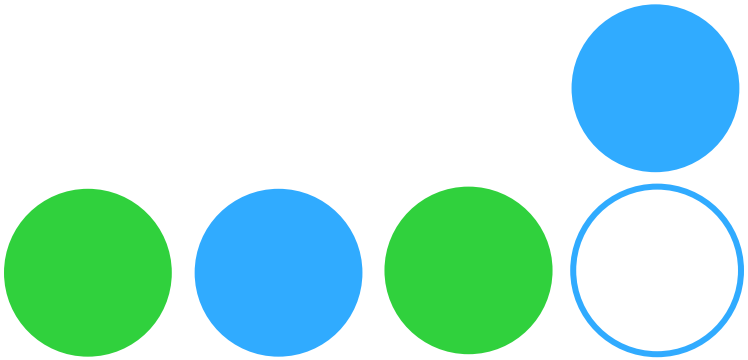
自己紹介

- 名前
  - 片山　暁雄（かたやまあきお）
- ID
  - id:c9katayama
- 所属
  - 株式会社キャピタルアセットプランニング
  - T2プロジェクト

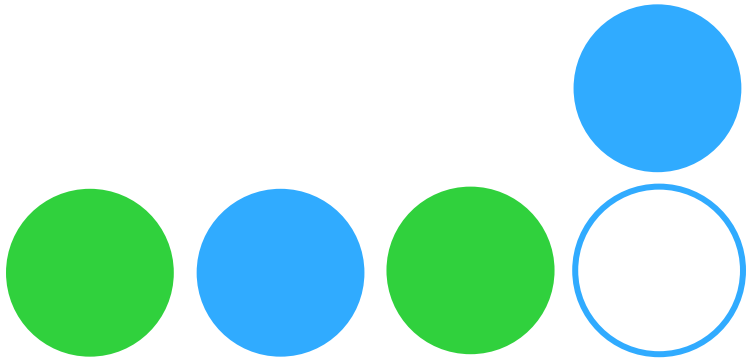


Agenda

- SDLoaderとは
- SDLoaderの生い立ち
- SDLoaderの特徴的機能
- シーン別利用法
- 案件事例
- 今後の展望
- まとめ

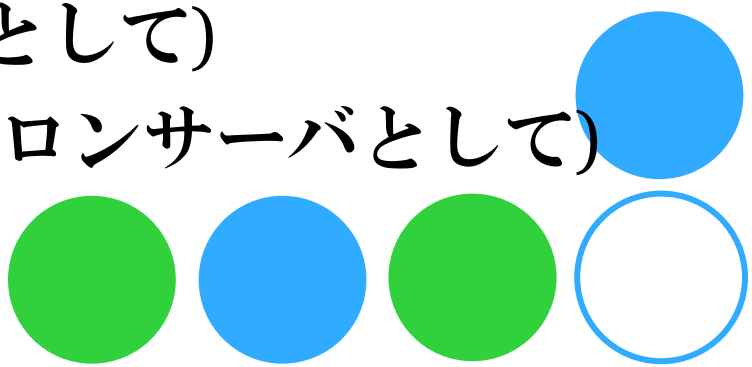


SDLoaderとは



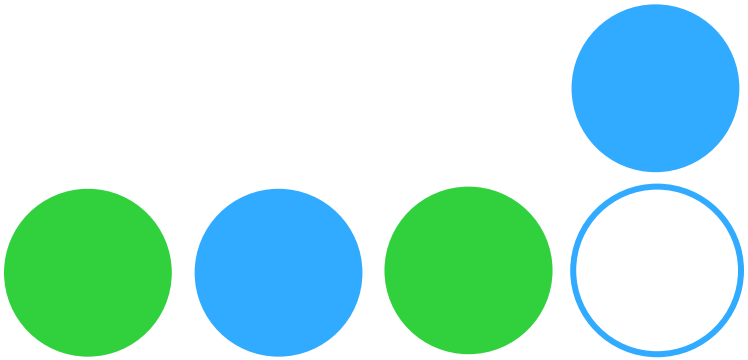
SDLoaderとは

- Webコンテナ
- ServletAPIを実装（2.4,2.5）
  - ただしgetPrincipal( )とかないです
- JSPも使える（2.0,2.1）
  - Jasperありがとう
- 基本Jar1個
  - newして起動(ライブラリとして)
  - exe,batで起動(スタンドアロンサーバとして)

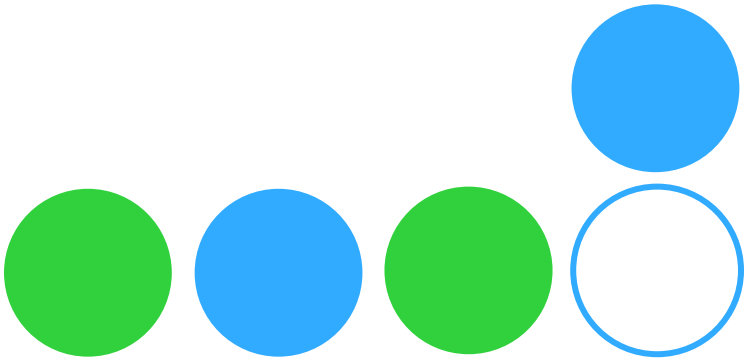


SDLoaderとは

- JDK5
- Apache2.0 License
- 現在v0_2_01

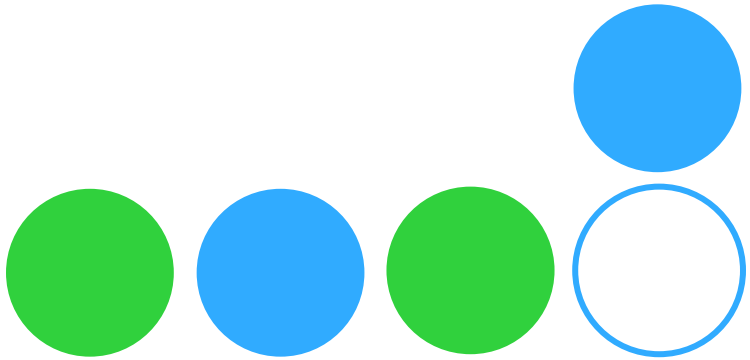


SDLoaderの生い立ち



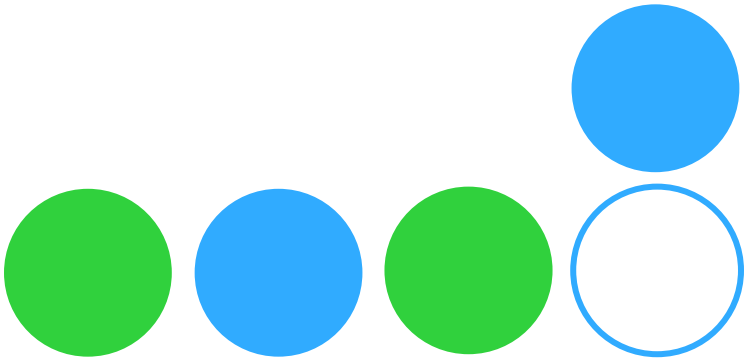
SDLoaderの生い立ち

- 元々はFlash-Javaのアプリケーションデモ用に作成
  - Servletだけ動けばよかった
- 営業や研修用に利用したい
  - スタンドアロン用途
  - 客先やネットワークがないところなどで
  - JSPも利用したい



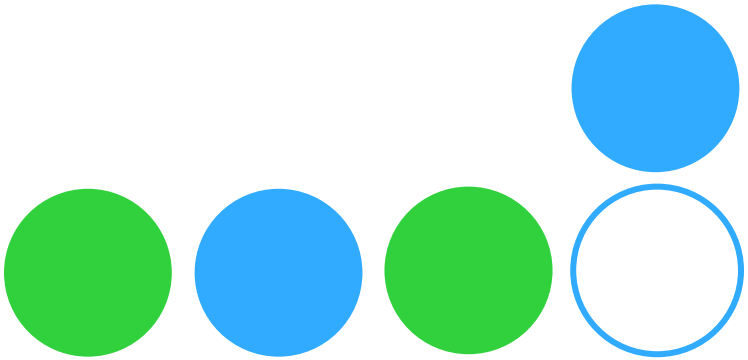
SDLoaderの生い立ち

- 利用者が楽に使えるもの
  - デモアプリがたくさんあるため、設定が面倒
  - ポート当たる　セキュリティエラー出る
  - 極力渡すファイルを少なくしたい
  - 軽くて早く起動

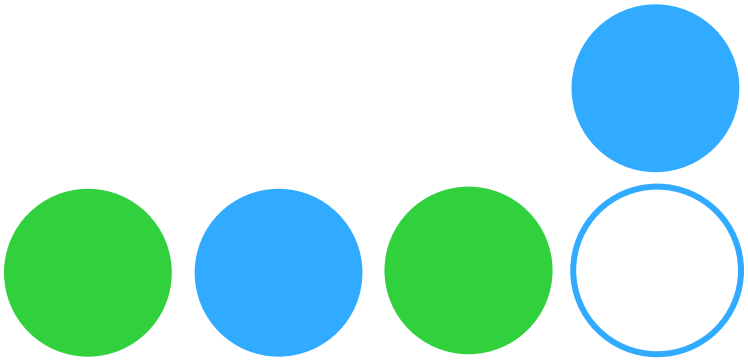


SDLoaderの生い立ち

- TomcatやWinstonも検討
  - Tomcatほど機能は要らない　手に負えない
  - Winstonもよかったけど、ライセンスがGPLだった
  - ServletAPIを実装してみたかった
- 現在
  - 開発に便利な機能を搭載
  - SWTとかの呼び出し

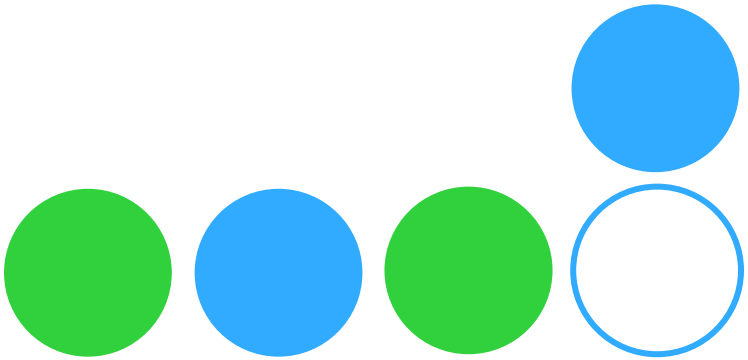


SDLoaderの特徴的機能



SDLoaderの特徴

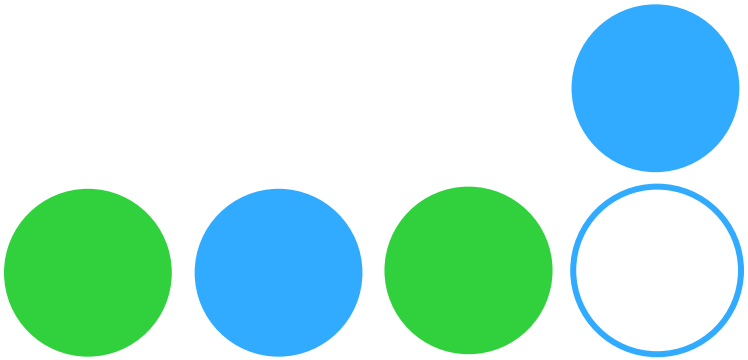
- Servletのみの最小jarが300K
  - リソースなしなら250K
  - JSP込みで2.4M
- newするWebコンテナ
  - プログラマブルに起動・停止可能。
  - Javaコードから各種設定やデプロイが可能



SDLoaderの特徴

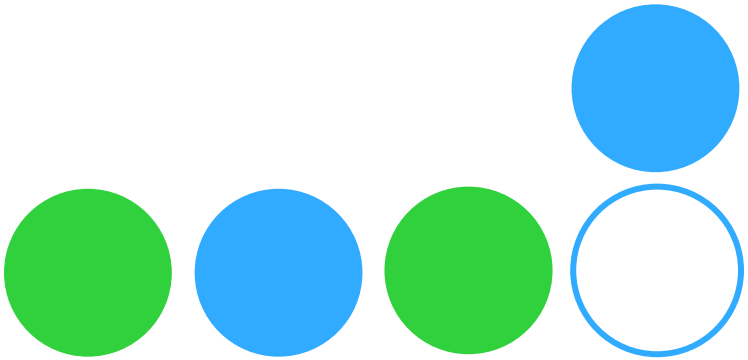
サンプル

```
//インスタンス化
SDLoader loader = new SDLoader(8080);

//WebApp追加
loader.addWebAppContext(
                new
WebAppContext("/sample","WebContent"));

//起動
loader.start();
```



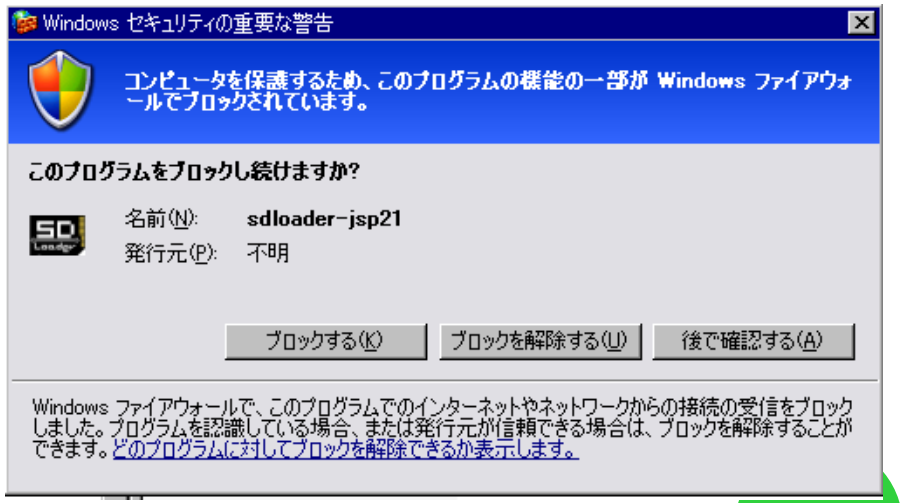
SDLoaderの特徴

- 利用ポート
  - デフォルト：30000　指定可能
  - 自動ポート検出
    - 指定ポートが使用中の場合、使えるポートを検出し使用
  - 管理ポート等はなし
    - ポートは１つのみ利用



SDLoaderの特徴

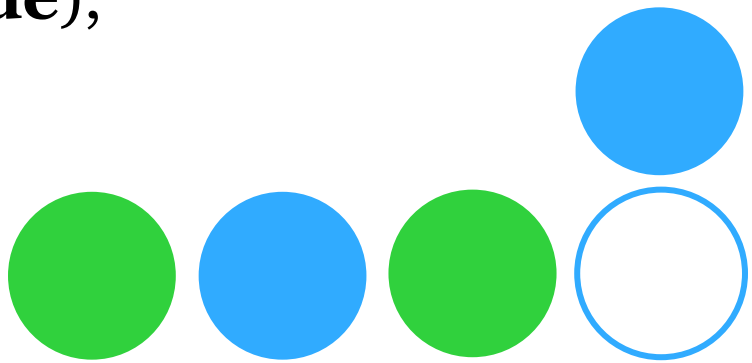
- ServerSocket
  - デフォルトでは、localhostのみListen
    - セキュリティエラーが出ない
    - 外向けポートをListenすることも可能



SDLoaderの特徴

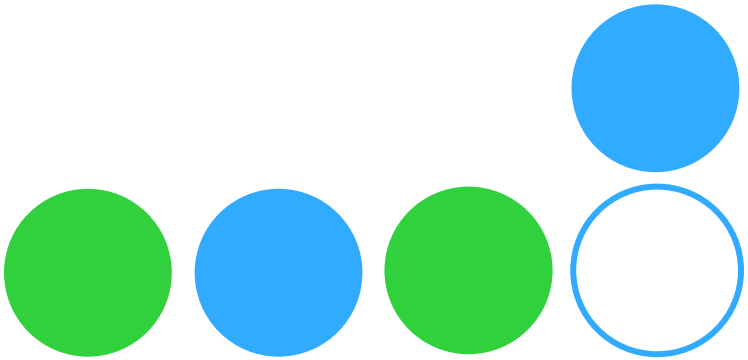
サンプル

```
//インスタンス化
SDLoader loader = new SDLoader(8080);

//自動ポート探知を使用
loader.setAutoPortDetect(true);

//外部ポートを使用
loader.setUseOutSidePort(true);
```

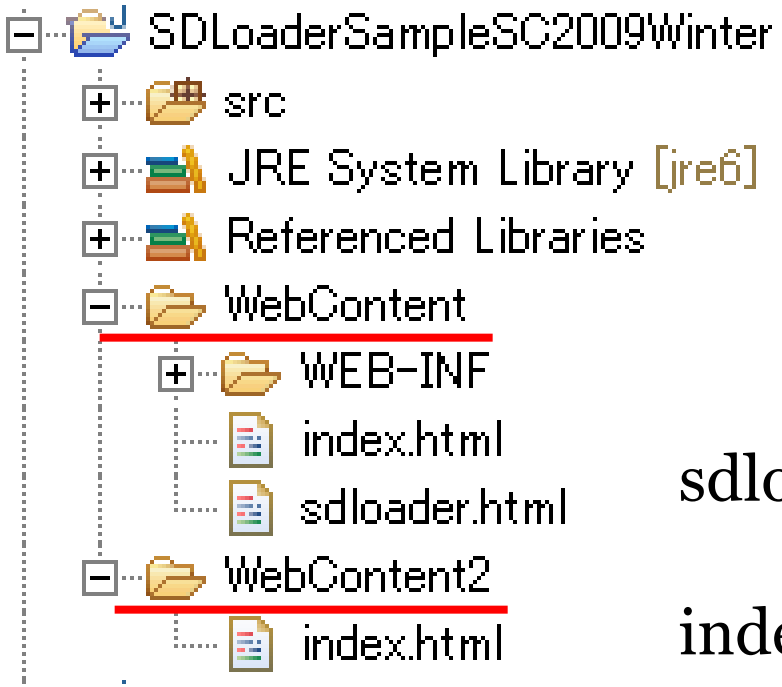


SDLoaderの特徴

- マルチドキュメントルート
  - １つのWebアプリに対して、複数のドキュメントルートを指定可能
  - リソース・クラスを複数の場所からロード



SDLoaderの特徴



sdloader.html・・・WebContentから

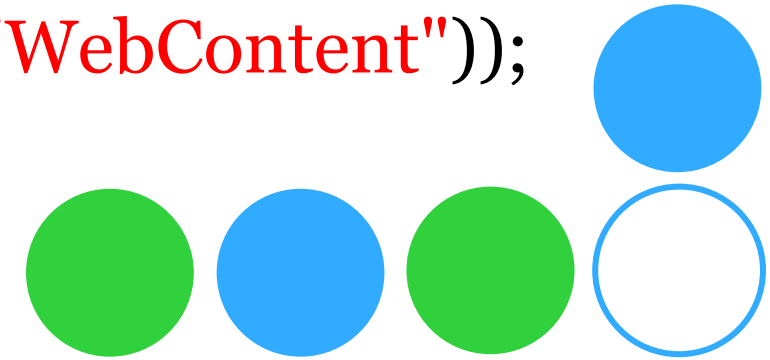
index.html・・・WebContent2から

```
// WebApp追加 ルートを複数指定
loader.addWebAppContext(
        new WebAppContext("/sample",
        "WebContent2","WebContent"));
```

SDLoaderの特徴

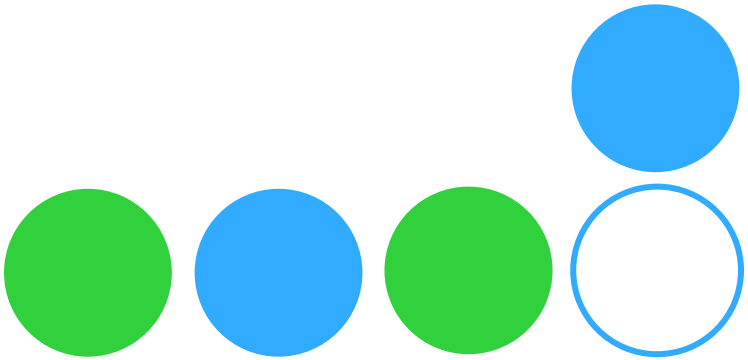
- ドキュメントルートは、絶対パス指定もしくはJava実行ディレクトリからの相対パス
- たとえば以下のコードで、隣のプロジェクトを指定したり出来ます

```
loader.addWebAppContext(
        new WebAppContext("/sample",
            “../Project2/WebContent",
            "WebContent"));
```



SDLoaderの特徴

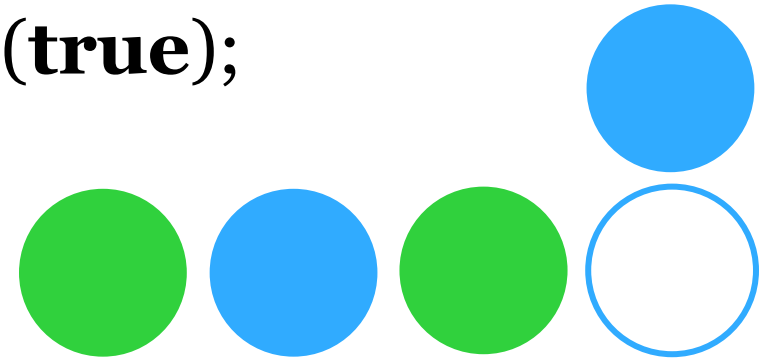
- classes,lib内のjarも、すべてクラスパスに通します
- web.xmlは、最初に見つけたものを使用



SDLoaderの特徴

- NoCache機能
  - ONにすると、すべてのレスポンスにNo-Cacheヘッダーをつけます
  - また、すべてのリクエストのlast-modified-sinceを無視します

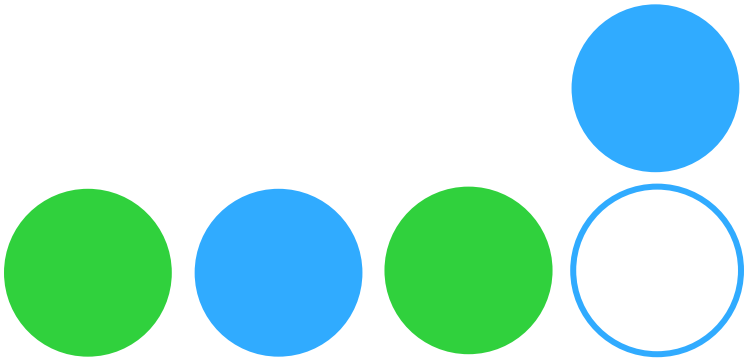
```
SDLoader loader = new SDLoader();
//NoCache機能を使用
loader.setUseNoCacheMode(true);
```



SDLoaderの特徴

- 帯域制限
  - 回線速度を擬似的に再現
  - 上り下りの両方に適用

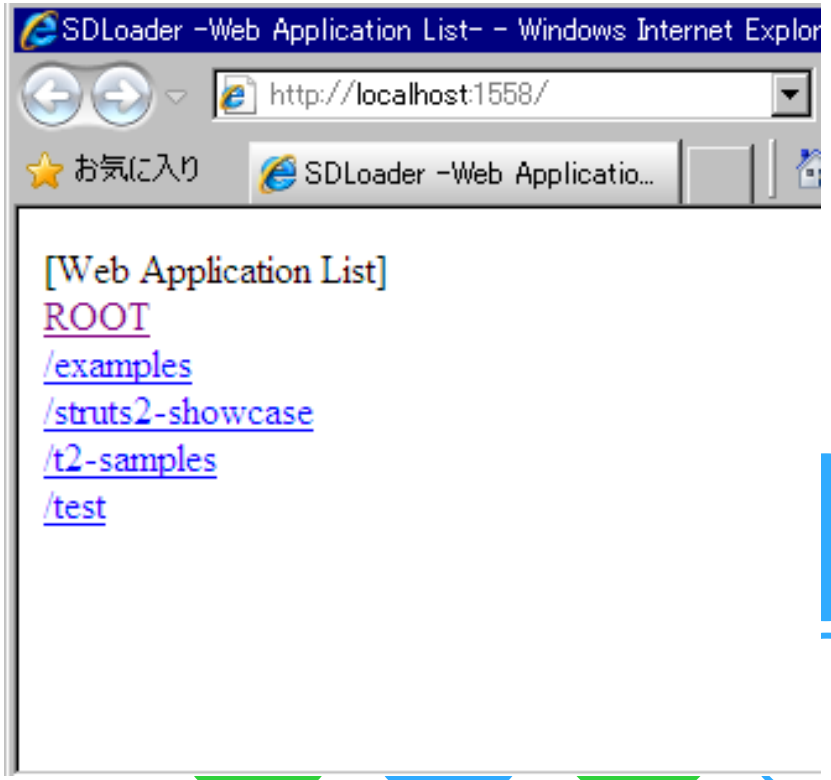
```
SDLoader loader = new SDLoader(8080);

//回線速度を設定
loader.setLineSpeed(LineSpeed.ISDN_64K_BPS);
```



SDLoaderの特徴

- デプロイ済みアプリ一覧
  - ルートURLにアクセスすると、デプロイ済みアプリの一覧を表示

各アプリのコンテキストルートへのリンク



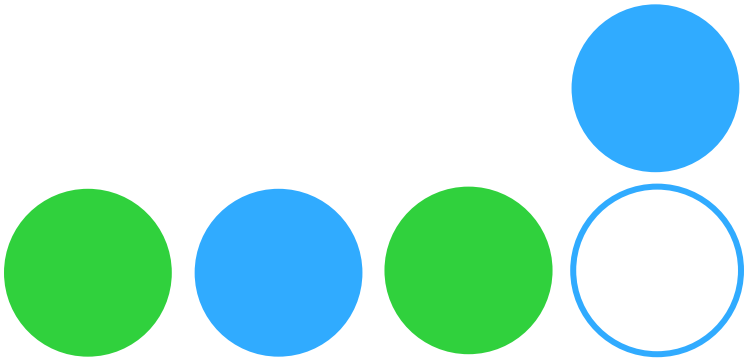
SDLoaderの特徴

- Browserクラス
  - 指定したURLをブラウザで開ける
  - OSのデフォルトブラウザを使用
    - IE,FF,Safari（Mac）で確認

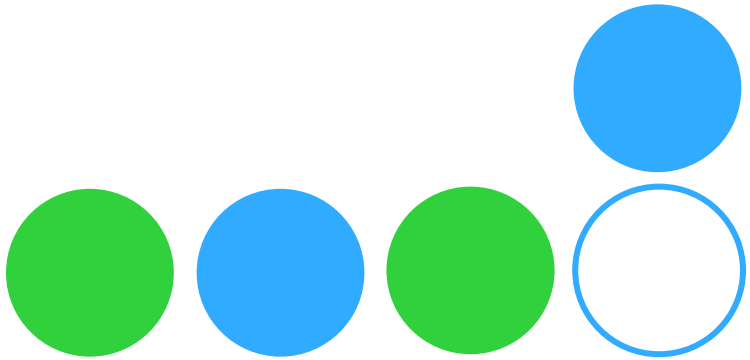
```
SDLoader loader = new SDLoader(8080);
loader.setAutoPortDetect(true);
loader.start();
//ブラウザで開く
Browser.open(
"http://localhost:"+loader.getPort()+"/sample/index.html")
;
```

SDLoaderの特徴

- プロジェクトテンプレート作成
  - Servlet24ProjectTemplateTool
  - Servlet25ProjectTemplateTool
  - 実行ディレクトリに、WEB-INFやweb.xmlの雛形を作成

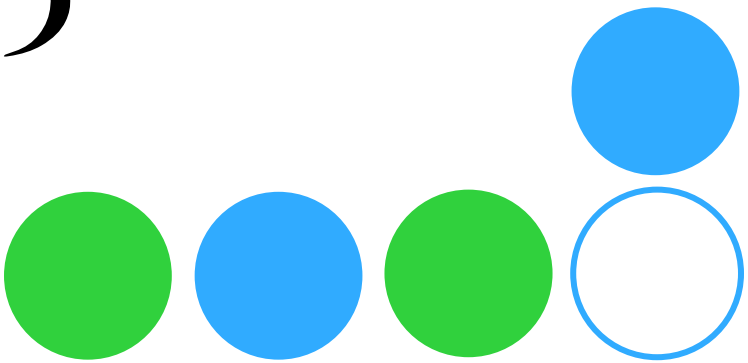


シーン別利用法



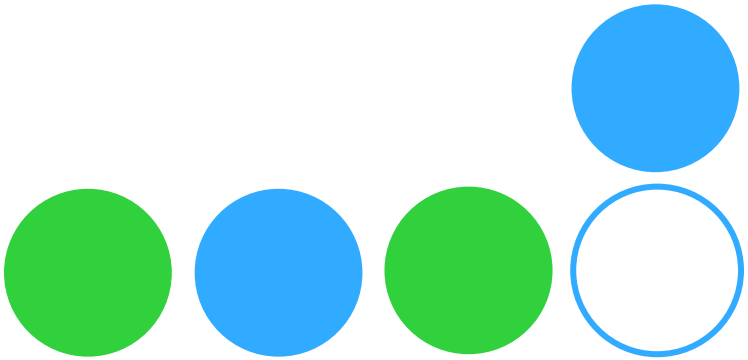
ケースその１

デモアプリ
作成依頼で
気を使う



デモアプリ作成依頼

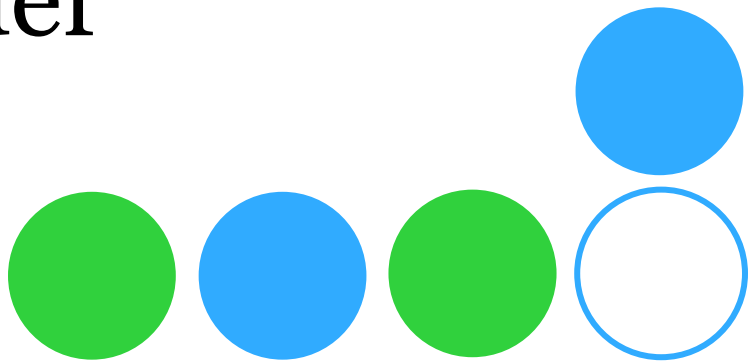
- 遠隔地のプロジェクト
- SpringとiBatisのサンプル作ってほしい
- 出来上がったらメールで送って



デモアプリ作成依頼

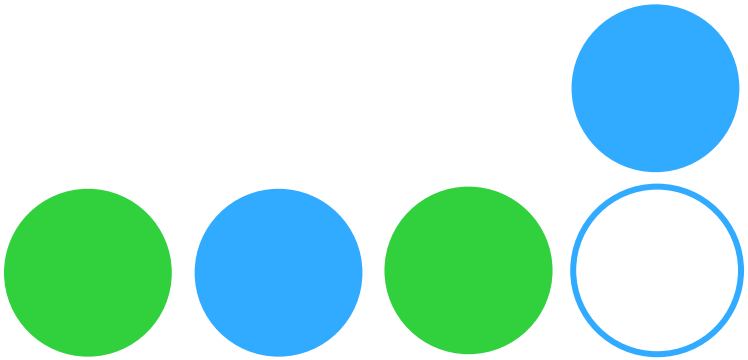
- TomcatPlugin?WTP?バージョンは？
- Eclipse ??
- デモ用設定を自分のTomcatに入れたくない
- 質問は来るからしばらくは置いとくけど、出来れば消したい

⇨ そこでSDLoader



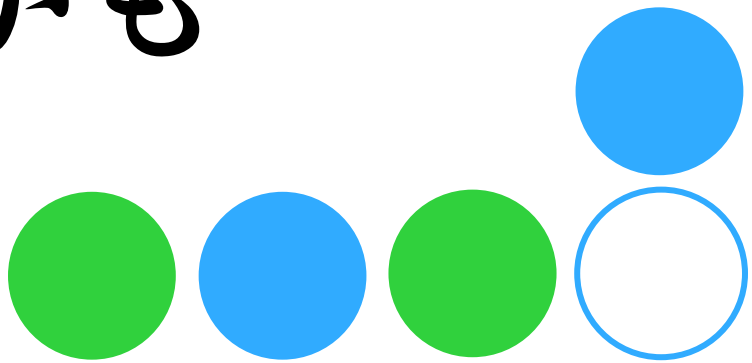
デモアプリ作成依頼

- 「とりあえずMainクラス実行して」
- Webコンテナのセッティング気にしない
- ブラウザも開けるようしておけば、説明の手間なし
- プロジェクトセットを取っておけば、消してもOK
- かならず動く安心感



ケースその２

試したい
だけど途中で
終わるかも



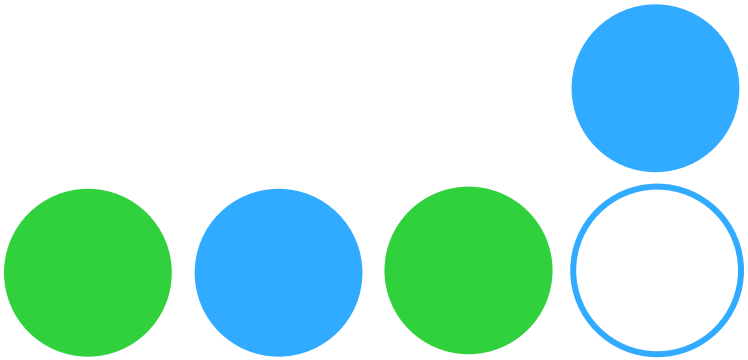
試しアプリ作成

- Webフレームワークの評価や、ちょっとしたアプリ
- とりあえず手っ取り早く動かしたい
- でも途中でやめるかも

⇒ そこでSDLoader

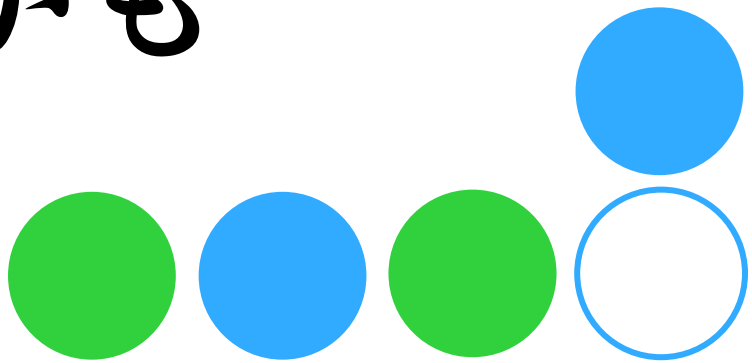
試しアプリ作成

- プロジェクトのファイルセットをSDLoaderごとどこかにおいておけば、作業再開可能
- Tomcatがエラーになる心配なし



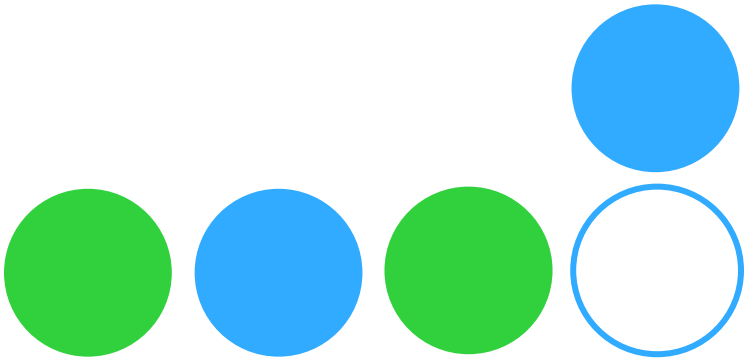
ケースその３

プロジェクト
分けたら毎回
コピーかも

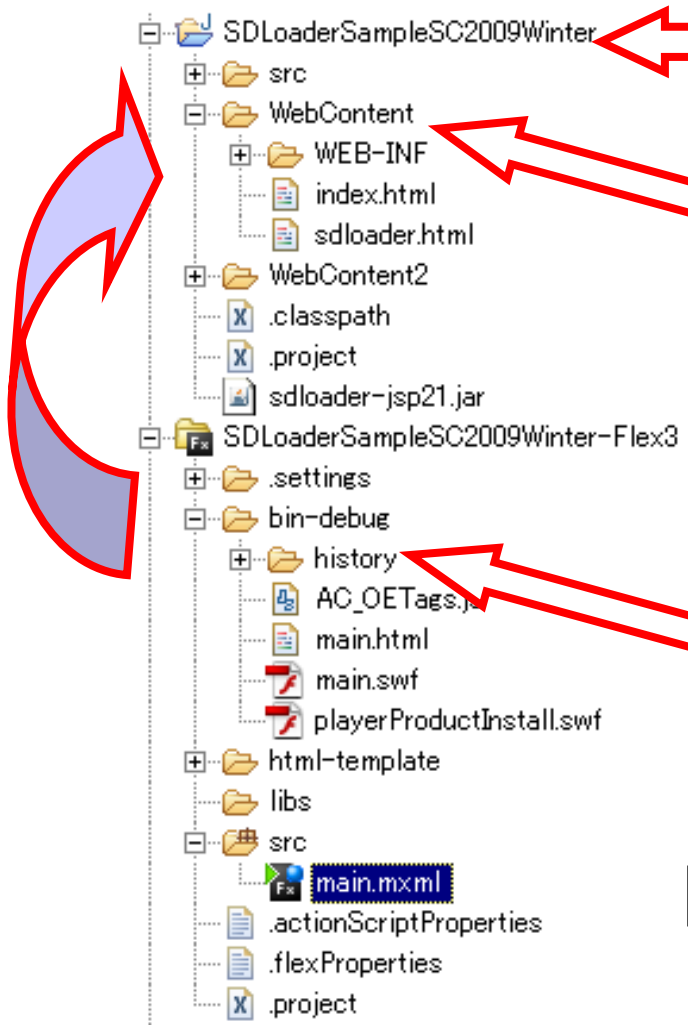


プロジェクト分け

- サーバ通信のあるFlexアプリケーション
- FlexプロジェクトとJavaプロジェクトを分けたい
- 動かすには、Javaプロジェクトにコンパイル済みのFlashファイル（swf）をコピーしないといけない



プロジェクト分け



Javaプロジェクト（Webアプリ）

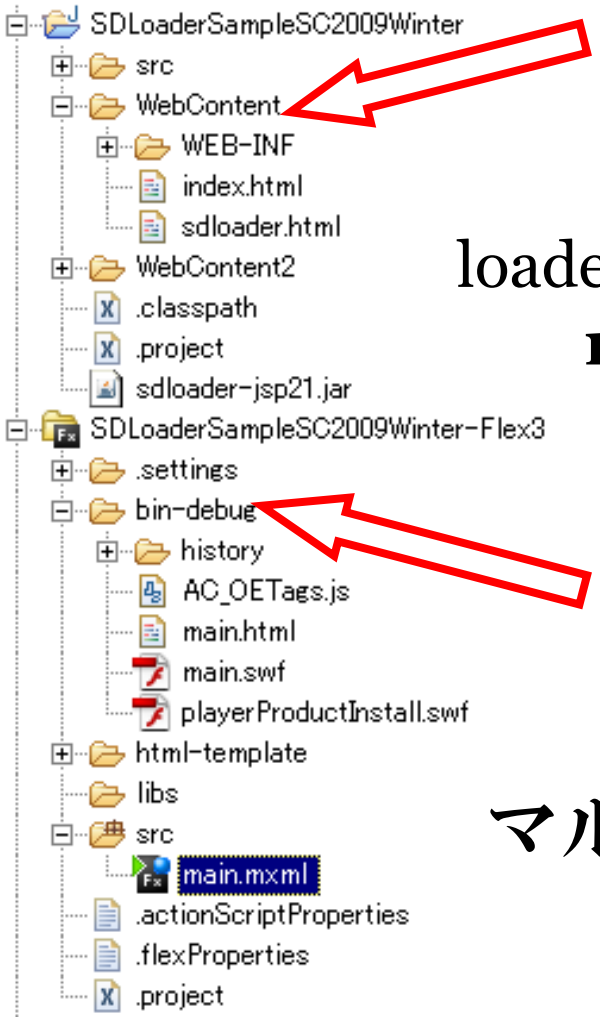
ダウンロードできるディレクトリ

Flexプロジェクト

Flashファイル出力先

そこでSDLoader

プロジェクト分け

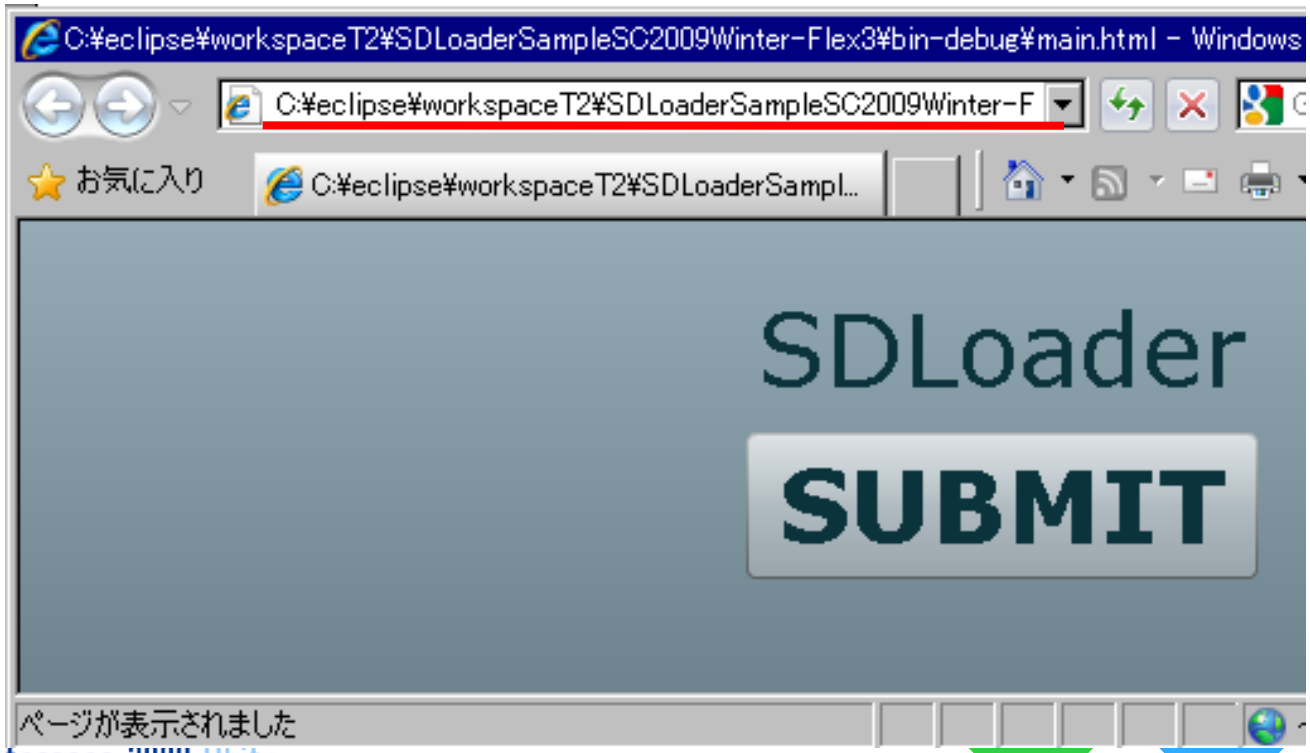



```
loader.addWebAppContext(
    new WebAppContext("/sample",
        "../SDLoaderSampleSC2009
            Winter-Flex3/bin-debug",
        "WebContent"));
```

マルチコンテキストルートで解決！

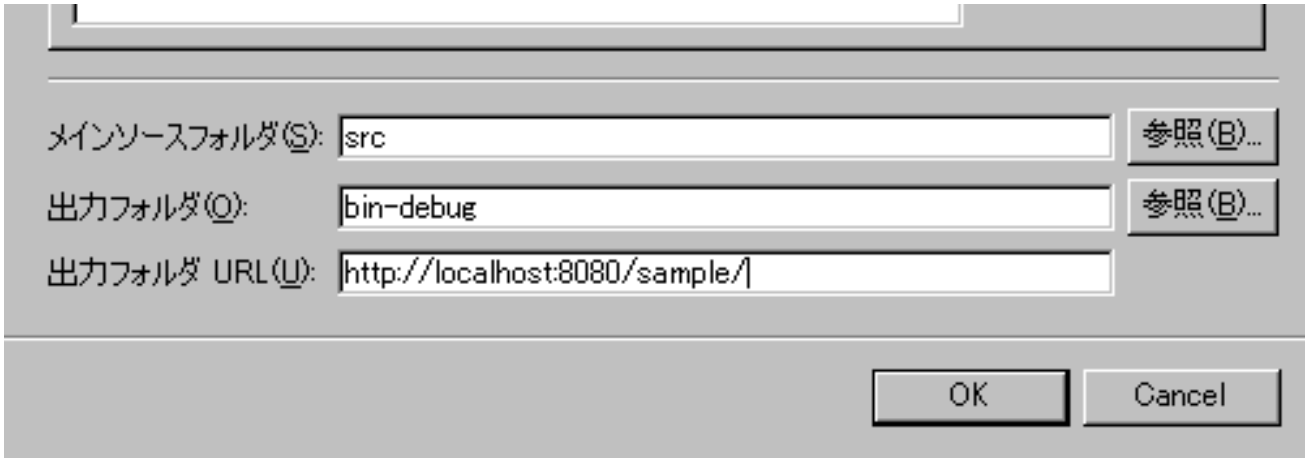
プロジェクト分け

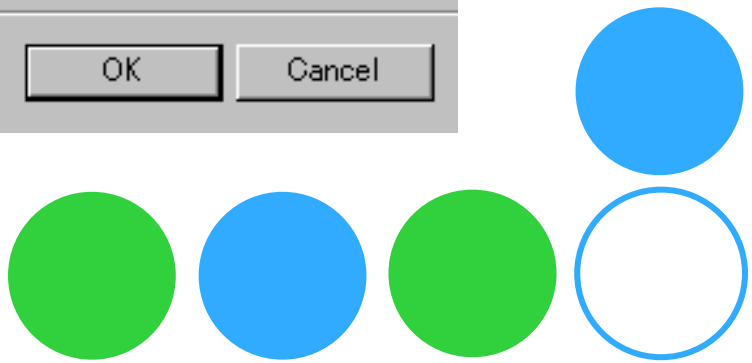
Flexプロジェクトはデフォルトだと、
ローカルファイルシステムを見に行く
→サーバ通信できない



プロジェクト分け

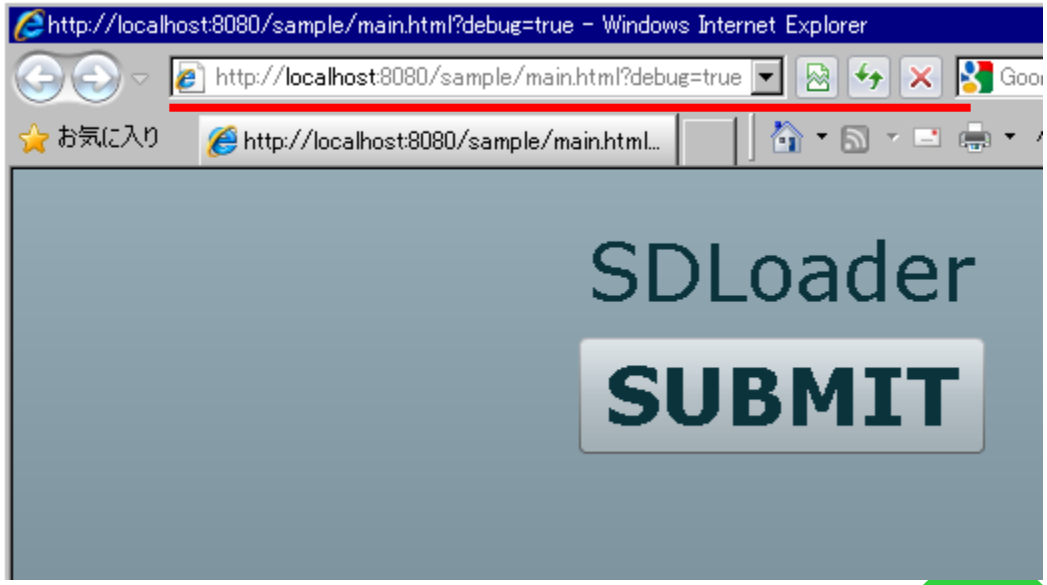
プロジェクトプロパティ
→Flexビルドパス
→出力フォルダURLを設定
→実行時にURLにアクセス





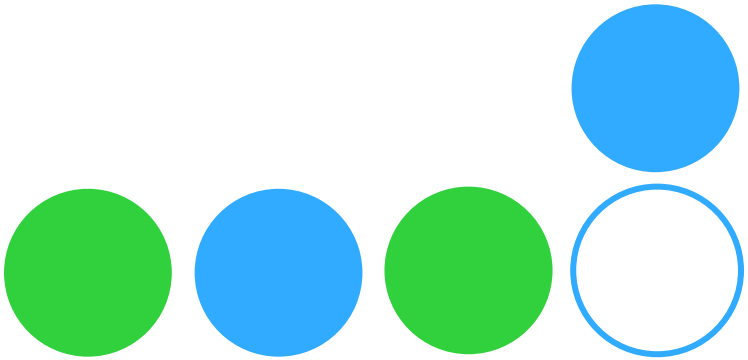
プロジェクト分け

Flex実行時も、通常のURLを使用する
Java、Flexともにデバックモードで起動すれば、
双方ともデバッグ可能。



プロジェクト分け

- No-Cacheモードにしておく
- プログレスバー開発には帯域制限

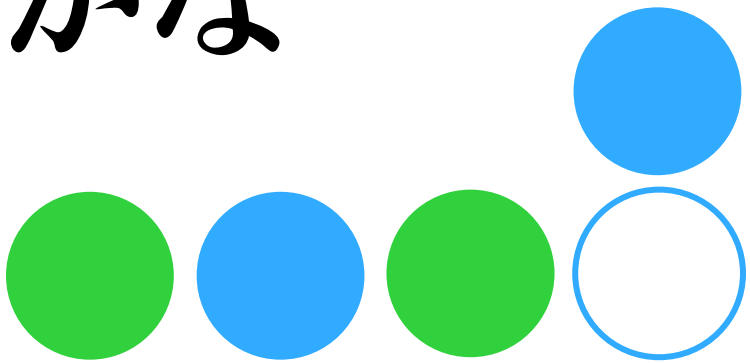


ケースその４

デモ利用

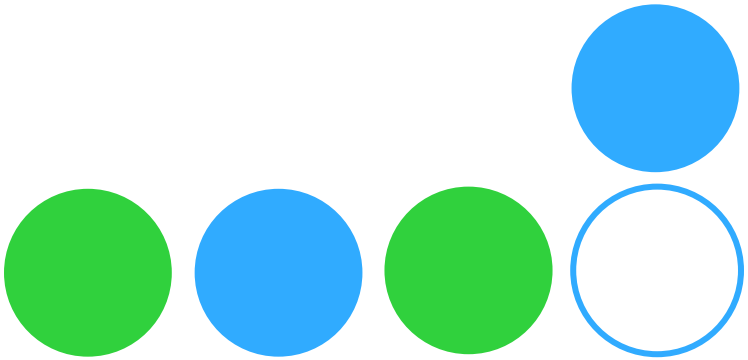
配布の決め手は

メディアかな



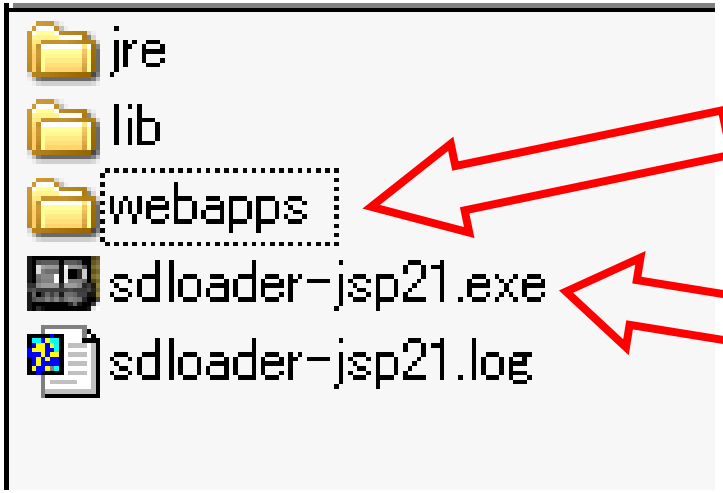
デモ利用

- デモでWebアプリを見せたい
- 営業の人の端末にセットアップめんどい
  - 外出が多いので、開発の人と時間が合わない
- ポートのかぶりとか気になる
  - JREのバージョンとかも
- まさかのアプリレンタル



デモ利用

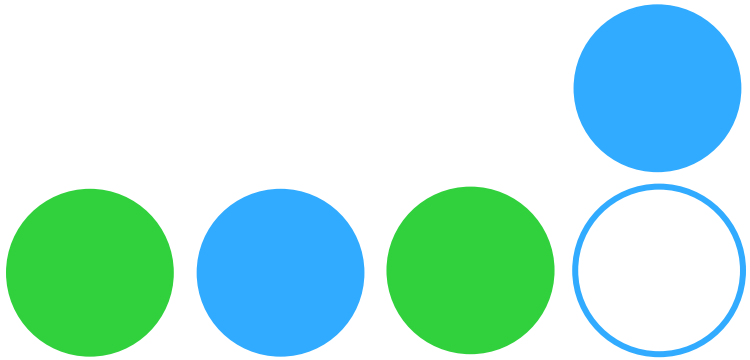
- ひとセットにしてCDやUSBに
- exe実行でサーバ起動ー＞ブラウザ立ち上げJREも同梱可能（exewrapの機能）



Warファイル入れる

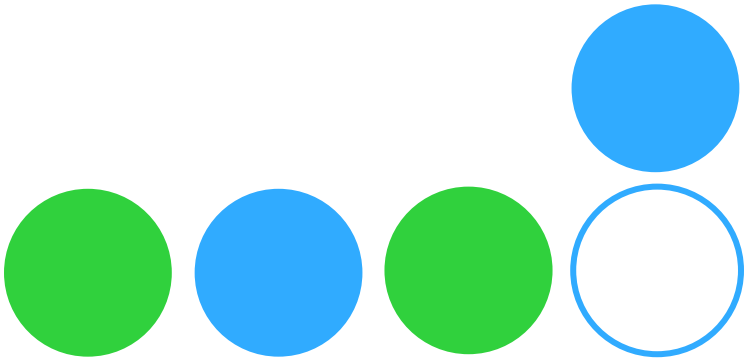
実行

案件事例



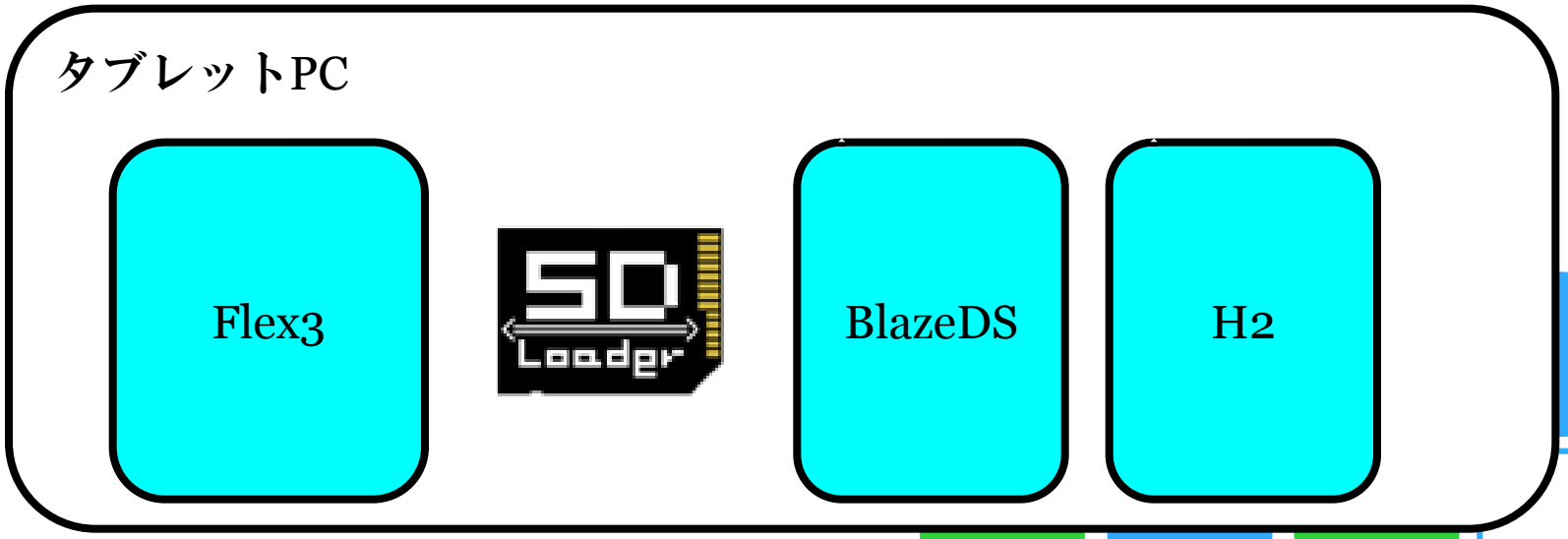
案件事例

- 本番アプリの研修用に、スタンドアロンで利用（Flash-Java,JSP-Java)
  - ネットのない研修室で利用
- 展示用にスタンドアロンで稼動(Flash-Java)



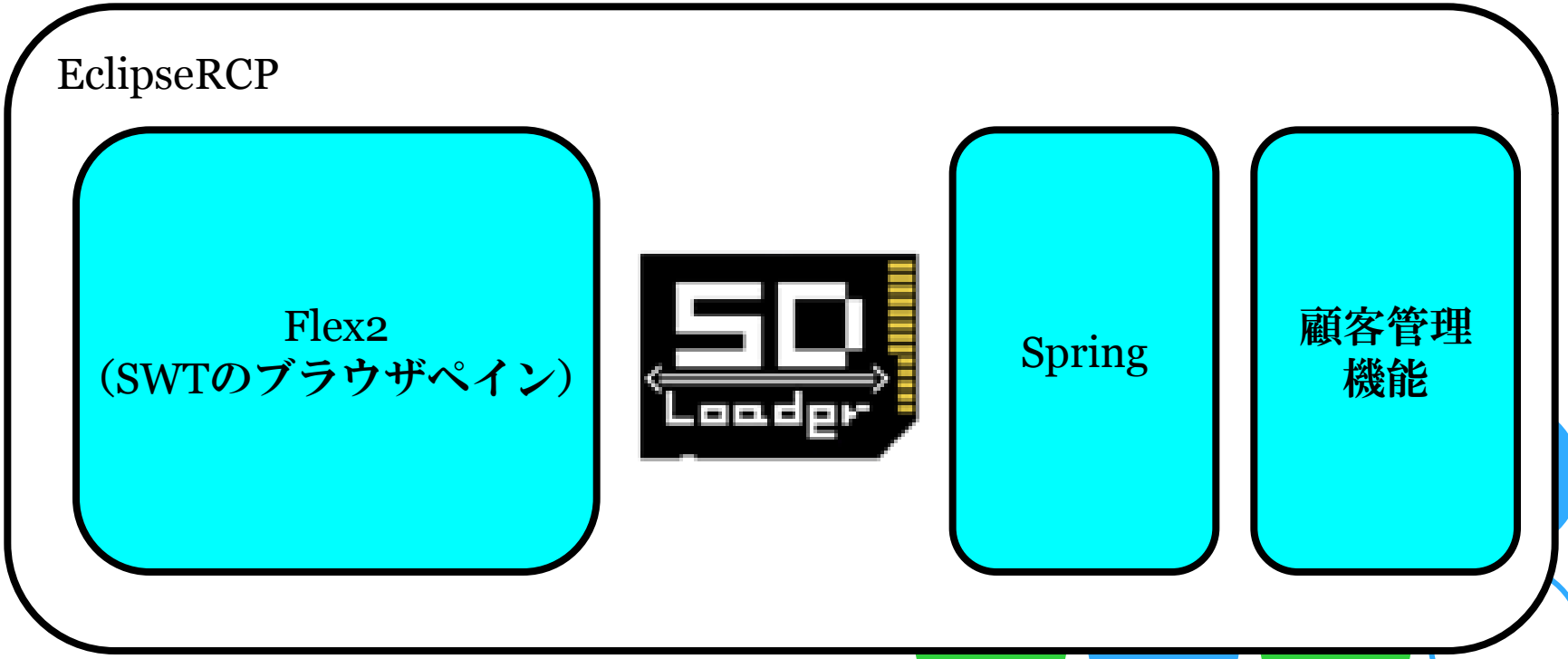
案件事例

- タブレットPCに入れ、スタンドアロンで利用(Flex3-Java)
  - 画面はFlex3できれいに
  - 複雑な計算、DBIO、帳票はJavaで

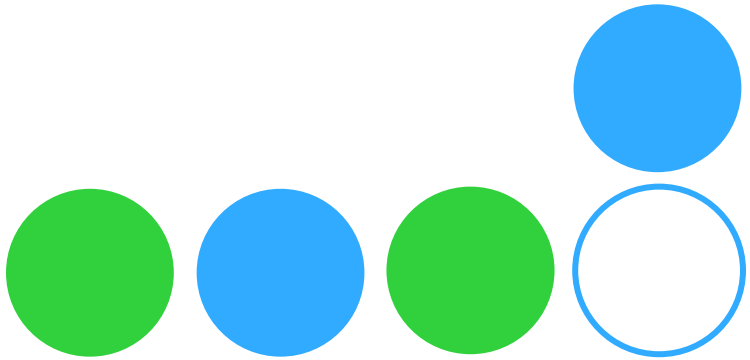


案件事例

- EclipseRCPで作成された顧客管理アプリに、Flex2のUIを搭載

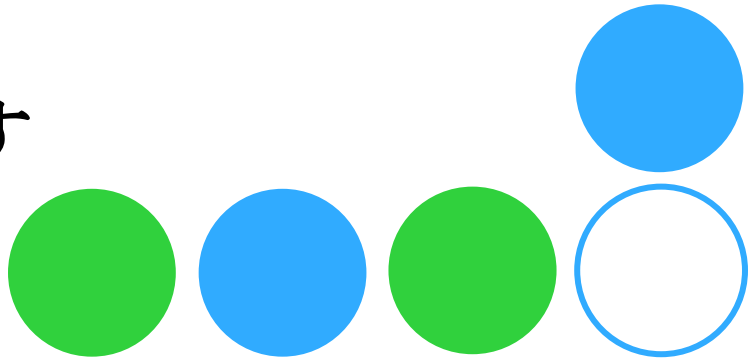


今後の展開

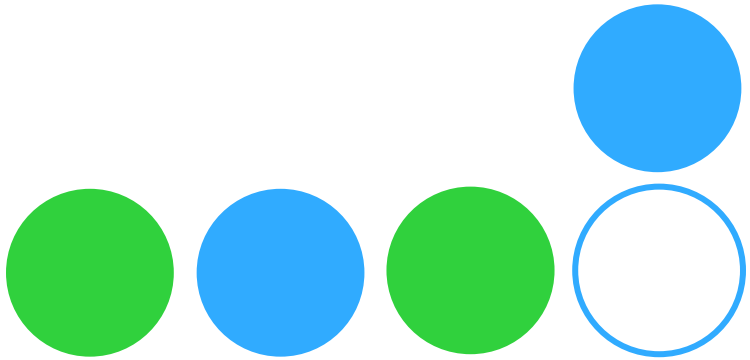


今後の展開

- アノテーション対応（@PreDestroyなど）
- JNDI対応
  - プログラマブルなJNDI
- 開発支援機能の充実
  - リクエストのリプレイ機能
  - キムキムホットデプロイ
- Cargoでテスト利用
  - 米林さんがやってくれます

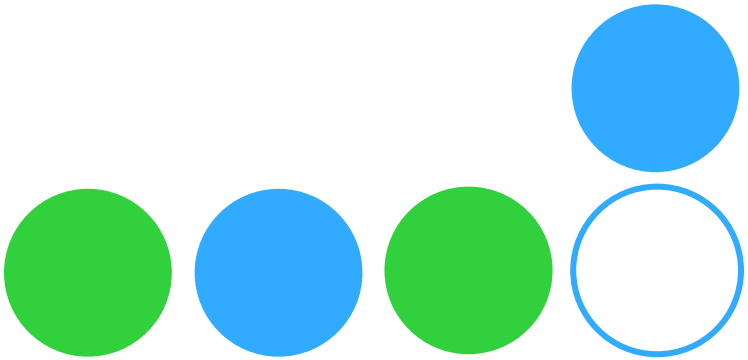


まとめ



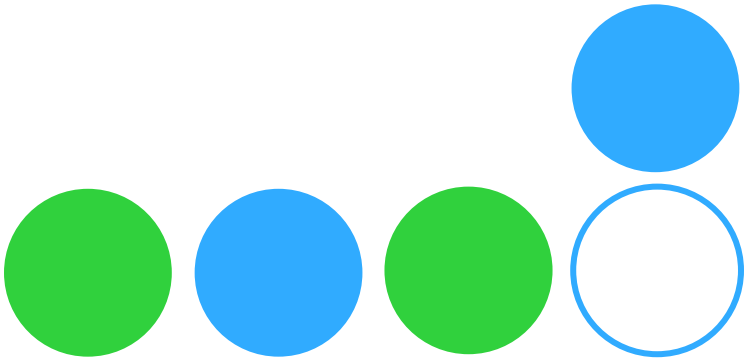
まとめ

- コンテナもいろいろな方法で利用できる！
- 開発用と割り切れば、規格おかまいなし
- ServletAPI実装はおもしろいしためになる



まとめ

- 開発リソース
  - サイト：　http://code.google.com/p/sdloader/
  - blog：　http://d.hatena.ne.jp/c9katayama/



御静聴ありがとうございました