

Control System Studio 3.0

プラグイン作成講座

Takashi Nakamoto
takashi.nakamoto@cosylab.com

Your **TRUSTED** Control System Partner



今日の主な目的



- CSS上で動作するアプリケーションを作成するための第一歩を理解する
 - CSSで何ができるのか
 - CSSの開発環境(Eclipse)の設定
 - 新しくプラグインを作成する方法
 - Channel AccessでPVから値を取得し、画面に表示する方法
 - 作成したプラグインを配布する方法

本日の予定



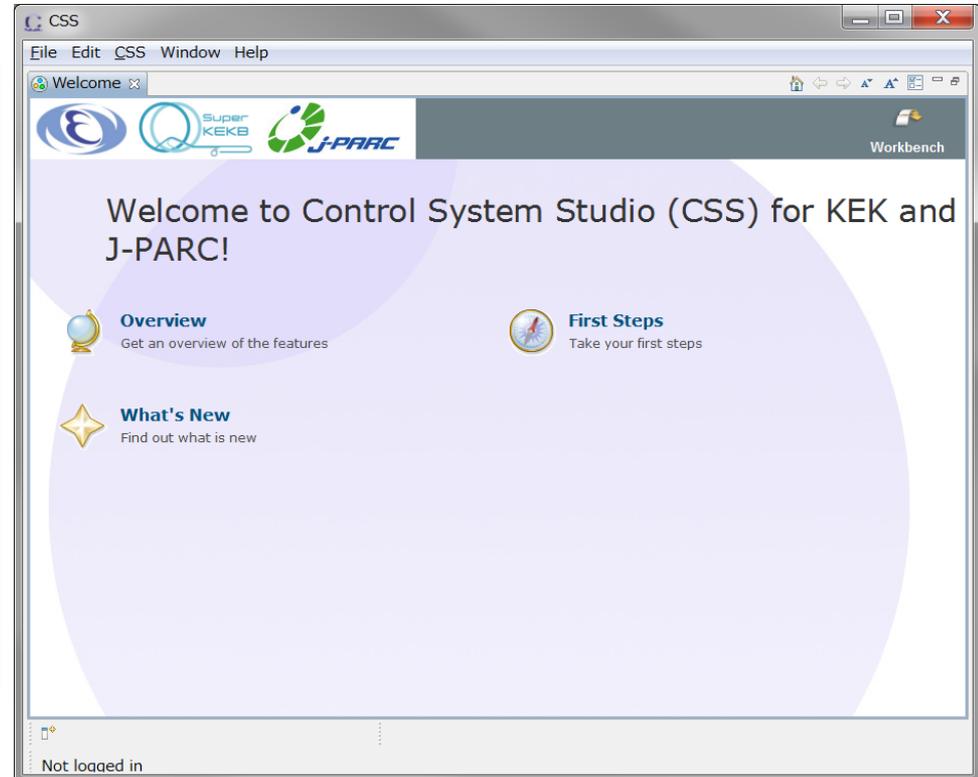
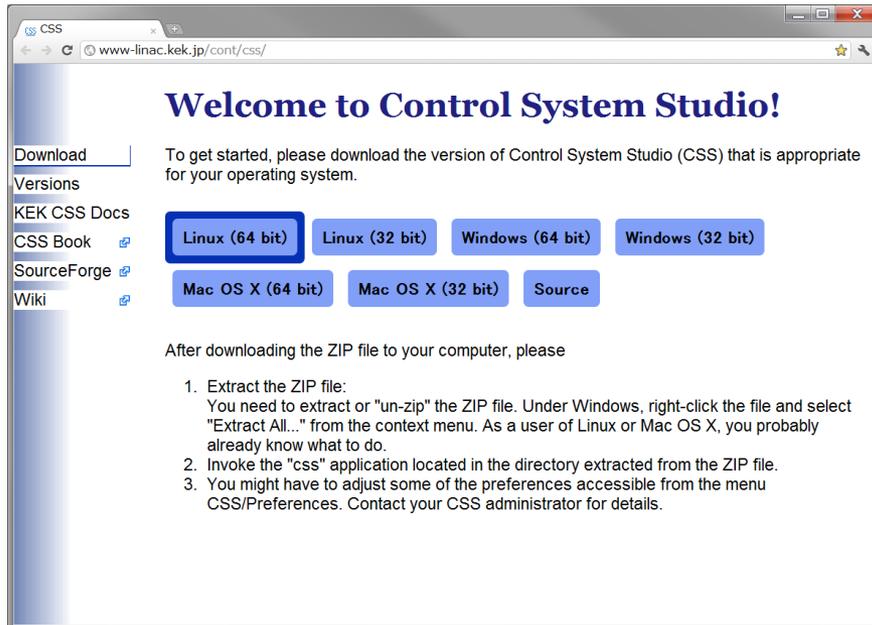
- 10:30 – 12:00
 - KEK版CSS
 - アプリケーション開発者にとってのCSS
 - 開発環境の概要
 - デモ:CSSの開発環境の設定
 - デモ:プラグインの作成
- 13:30 – 15:00
 - デモ:PVの取得と画面への描画
 - デモ:作成したアプリケーションの配布
- 15:00 –
 - 質疑応答

KEK版CSS

Your **TRUSTED** Control System Partner



□ <http://www-linac.kek.jp/cont/css/>



最近の開発成果



- 各加速器向けの起動スクリプト
 - EPICS Address Listの設定
 - アーカイバの設定
 - （今のところLINAC向けの設定のみ）
- Kblogの読み出し
 - DataBrowserなどで表示できるように

Control System Studio 3.0 プラグイン作成講座

アプリケーション 開発者にとってのCSS

アプリケーション開発者にとって COSYLAB

- CSSは複数のアプリケーションを協調させて動作させるプラットフォーム
 - スタンドアロンの複数のアプリケーションから、1つのウィンドウに統合されたアプリケーションへ
 - 例) アラーム、履歴表示

BOYも選択肢の一つ

□ BOYを使うことのメリット

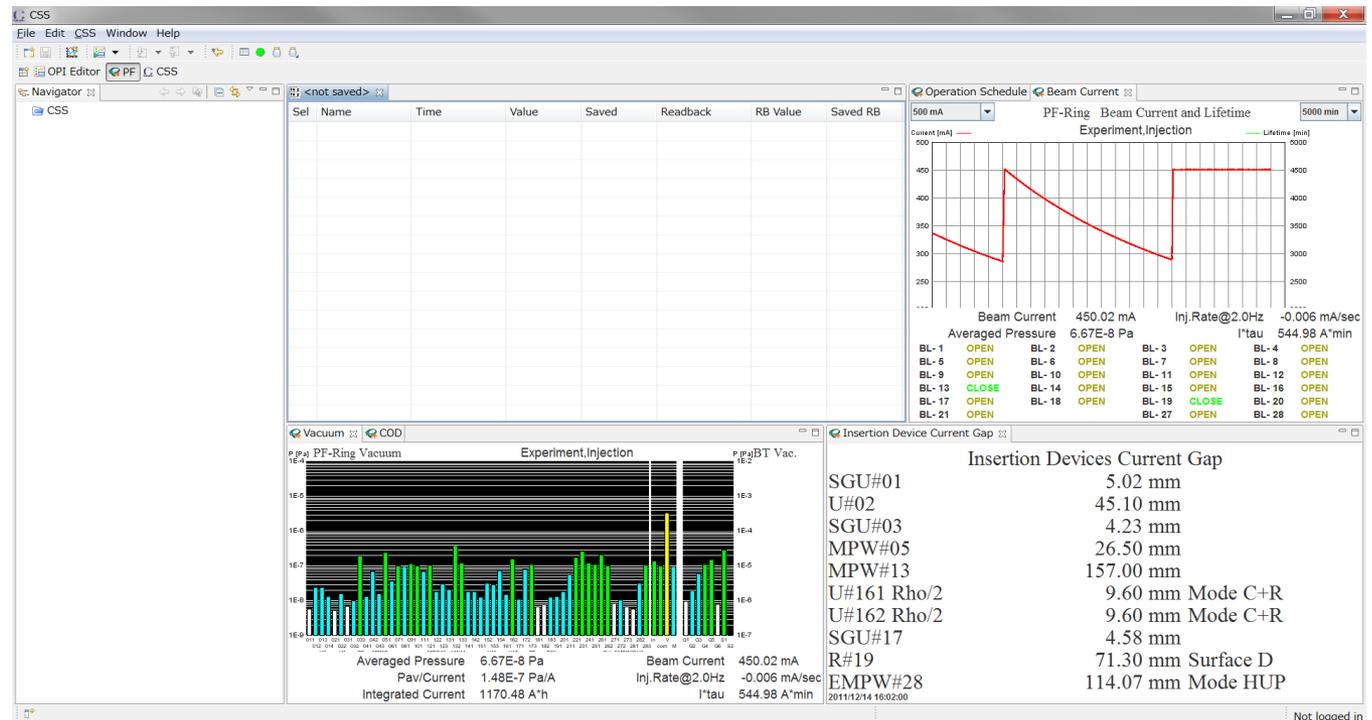
- Javaを覚える必要が無く、気軽にはじめられる
- BOYでできることはBOYでやったほうが簡単で早い
 - EDMやMEDMでできたこと
 - スクリプト(JavaScript or Python)を利用した画面描画
- BOY向けに新たにウィジェットを作ることも可能

□ BOYがあまり向かないことの例

- PVのみに基づかないスクリーン描画
 - 履歴データ
 - データベース
 - ファイル
- 多くの要素が絡む複雑なアニメーション
- 3D描画

統合環境としての可能性

- 既存のアプリケーションの活用
 - Javaアプレット
 - 静的なWebページ
 - Webアプリケーション
 - ActiveX



http://www-pfring.kek.jp/PF/status/java/java_index_j_html にあったアプレットを流用

Control System Studio 3.0 プラグイン作成講座

開発環境の概要

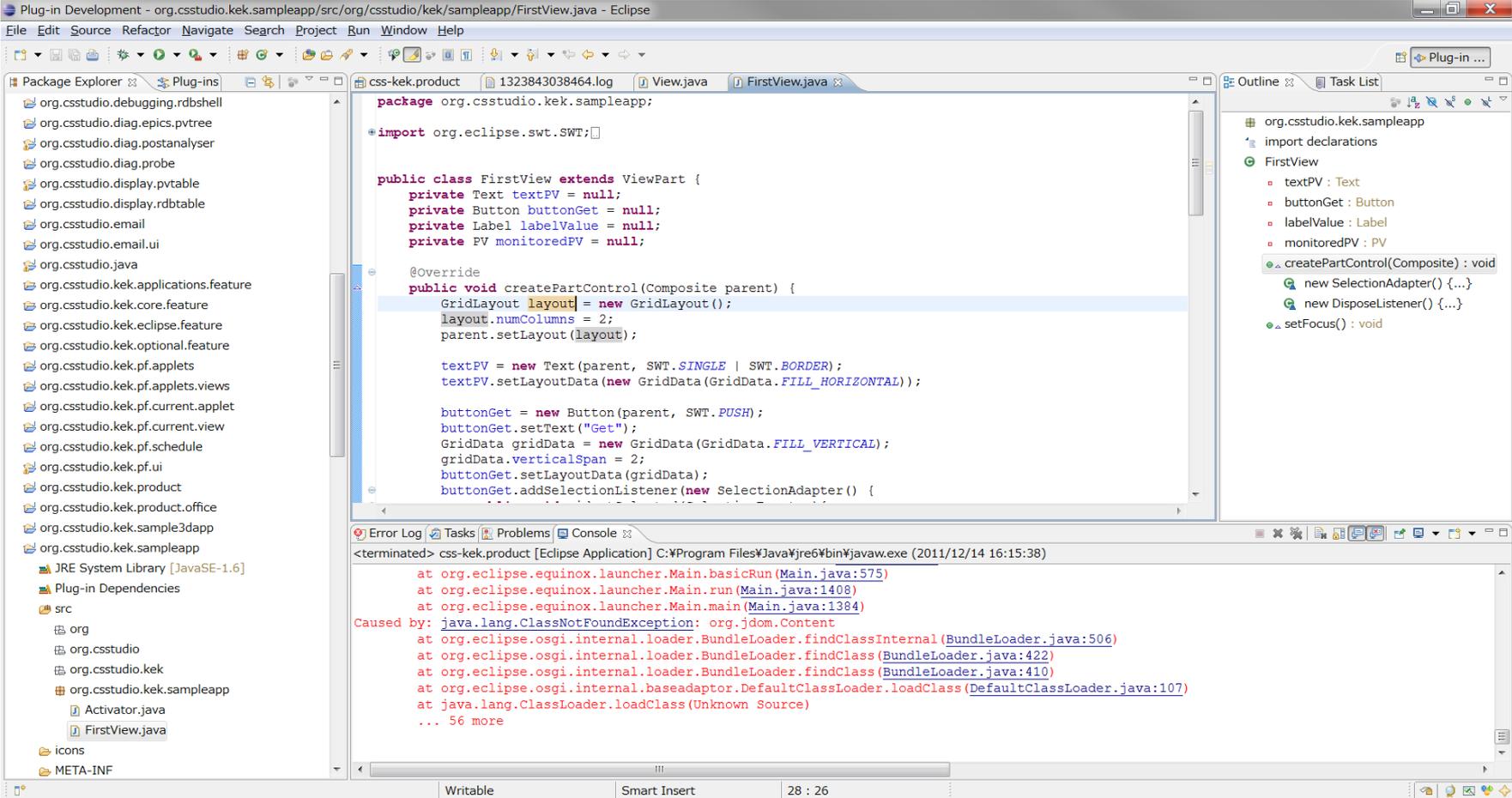
Eclipse RCPとは

Eclipse RCP

- ❑ RCP = Rich Client Platform
- ❑ Eclipse自身がEclipse RCPに基づいたアプリケーションの一つ
 - 開発環境であるEclipseの使い方をよく理解しておくことが、CSSのプラグイン開発にとっても役に立つ
 - ビュー
 - エディタ
 - パースペクティブ
 - プラグイン
 - フィーチャー
 - アップデートサイト
 - 設定
 - ヘルプ

Eclipseのウィンドウ

Eclipseの画面構成要素



The screenshot displays the Eclipse IDE interface. The Package Explorer on the left shows the project structure for 'org.csstudio.kek.sampleapp'. The main editor shows the code for the 'FirstView' class, which extends 'ViewPart'. The code includes imports for 'org.eclipse.swt.SWT' and defines several private fields: 'textPV', 'buttonGet', 'labelValue', and 'monitoredPV'. The 'createPartControl' method is overridden, creating a 'GridLayout' and adding a 'Text' widget and a 'Button' widget. The Error Log at the bottom shows a 'ClassNotFoudException' error, which is a typo for 'ClassNotFoundException'. The error message indicates that the class 'org.jdom.Content' could not be found, and it lists the stack trace starting from 'Main.java:575'.

```

package org.csstudio.kek.sampleapp;

import org.eclipse.swt.SWT;

public class FirstView extends ViewPart {
    private Text textPV = null;
    private Button buttonGet = null;
    private Label labelValue = null;
    private PV monitoredPV = null;

    @Override
    public void createPartControl(Composite parent) {
        GridLayout layout = new GridLayout();
        layout.numColumns = 2;
        parent.setLayout(layout);

        textPV = new Text(parent, SWT.SINGLE | SWT.BORDER);
        textPV.setLayoutData(new GridData(GridData.FILL_HORIZONTAL));

        buttonGet = new Button(parent, SWT.PUSH);
        buttonGet.setText("Get");
        GridData gridData = new GridData(GridData.FILL_VERTICAL);
        gridData.verticalSpan = 2;
        buttonGet.setLayoutData(gridData);
        buttonGet.addSelectionListener(new SelectionAdapter() {

```

Error Log:

```

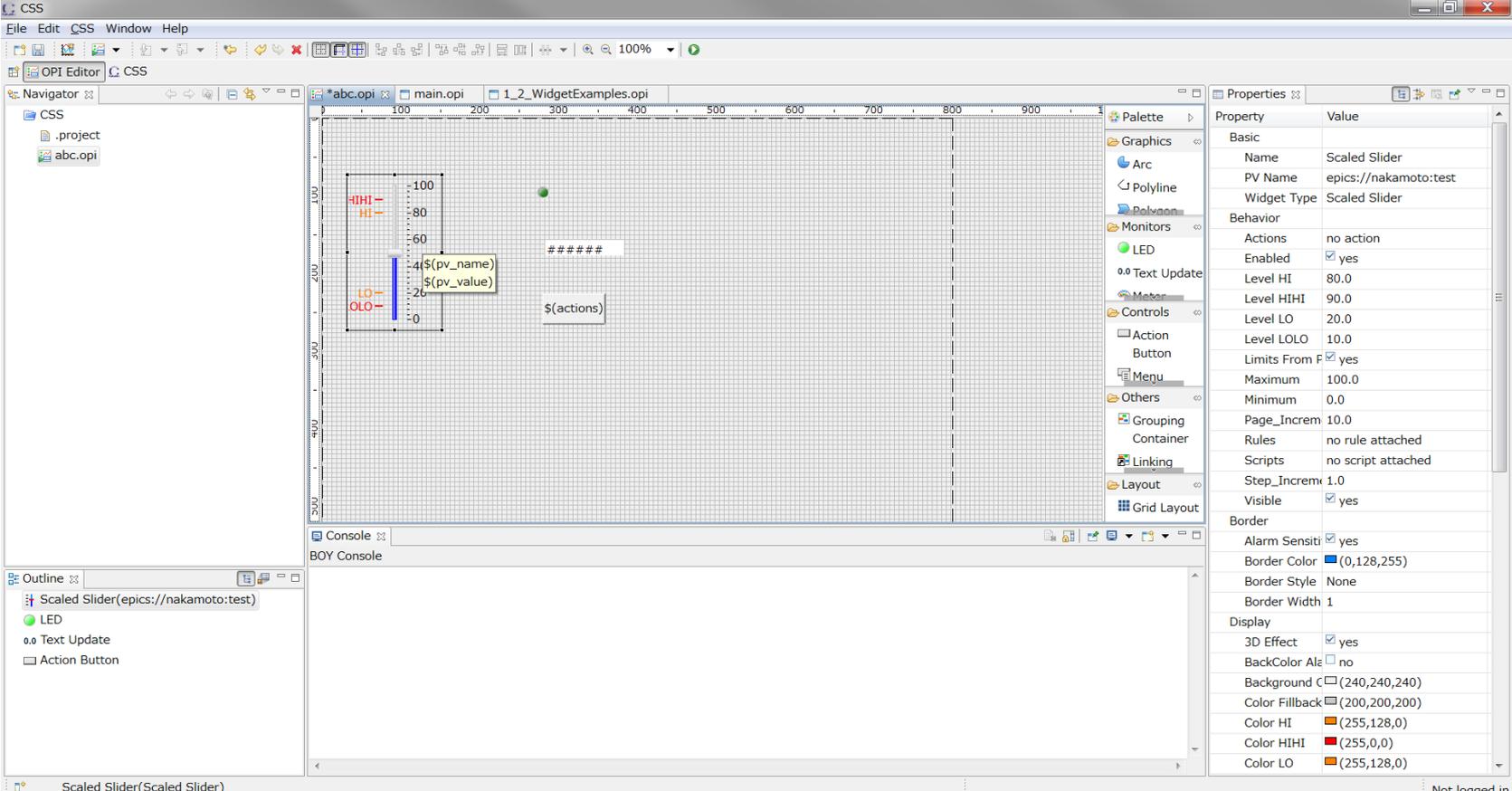
<terminated> css-kek.product [Eclipse Application] C:\Program Files\Java\jre6\bin\javaw.exe (2011/12/14 16:15:38)

    at org.eclipse.equinox.launcher.Main.basicRun(Main.java:575)
    at org.eclipse.equinox.launcher.Main.run(Main.java:1408)
    at org.eclipse.equinox.launcher.Main.main(Main.java:1384)
Caused by: java.lang.ClassNotFoundException: org.jdom.Content
    at org.eclipse.osgi.internal.loader.BundleLoader.findClassInternal(BundleLoader.java:506)
    at org.eclipse.osgi.internal.loader.BundleLoader.findClass(BundleLoader.java:422)
    at org.eclipse.osgi.internal.loader.BundleLoader.findClass(BundleLoader.java:410)
    at org.eclipse.osgi.internal.baseadaptor.DefaultClassLoader.loadClass(DefaultClassLoader.java:107)
    at java.lang.ClassLoader.loadClass(Unknown Source)
    ... 56 more

```

CSSのウィンドウ

□ CSSの画面構成要素



The screenshot displays the CSS OPI Editor interface. The main workspace shows a Scaled Slider widget with a vertical scale from 0 to 100. The widget is labeled with HI, HIHI, LO, and LOLO. The current value is 4, and the PV name is epics://nakamoto:test. The widget is connected to a BOY Console.

The Properties panel on the right shows the configuration for the Scaled Slider widget:

Property	Value
Basic	
Name	Scaled Slider
PV Name	epics://nakamoto:test
Widget Type	Scaled Slider
Behavior	
Actions	no action
Enabled	<input checked="" type="checkbox"/> yes
Level HI	80.0
Level HIHI	90.0
Level LO	20.0
Level LOLO	10.0
Limits From F	<input checked="" type="checkbox"/> yes
Maximum	100.0
Minimum	0.0
Page_Increm	10.0
Rules	no rule attached
Scripts	no script attached
Step_Increm	1.0
Visible	<input checked="" type="checkbox"/> yes
Border	
Alarm Sensiti	<input checked="" type="checkbox"/> yes
Border Color	(0,128,255)
Border Style	None
Border Width	1
Display	
3D Effect	<input checked="" type="checkbox"/> yes
BackColor Ale	<input type="checkbox"/> no
Background C	(240,240,240)
Color Fillback	(200,200,200)
Color HI	(255,128,0)
Color HIHI	(255,0,0)
Color LO	(255,128,0)

The Outline panel on the left shows the widget hierarchy:

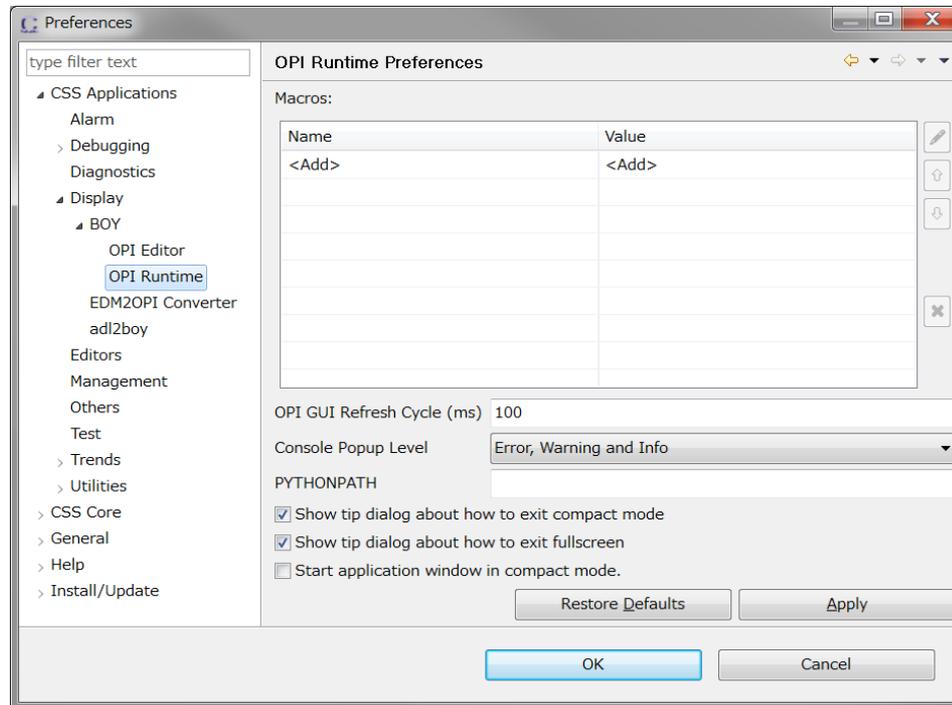
- Scaled Slider(epics://nakamoto:test)
 - LED
 - Text Update
 - Action Button

画面の構成要素



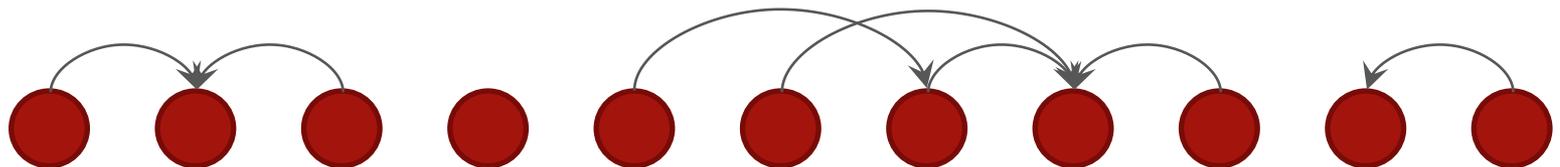
- ワークベンチ
 - ウィンドウ全体のこと
- メニュー
- ツールバー
- エディタ
 - データの作成、編集、保存を行うための領域
 - 例) DataBrowser, OPI Editor
- ビュー
 - 情報を表示するための領域
 - 例) Probe, Alarm Tree View
- パースペクティブ
 - エディタ、ビュー、メニュー、ツールバーの組み合わせと配置

- アップデートサイト
- 設定
 - 統一された設定画面



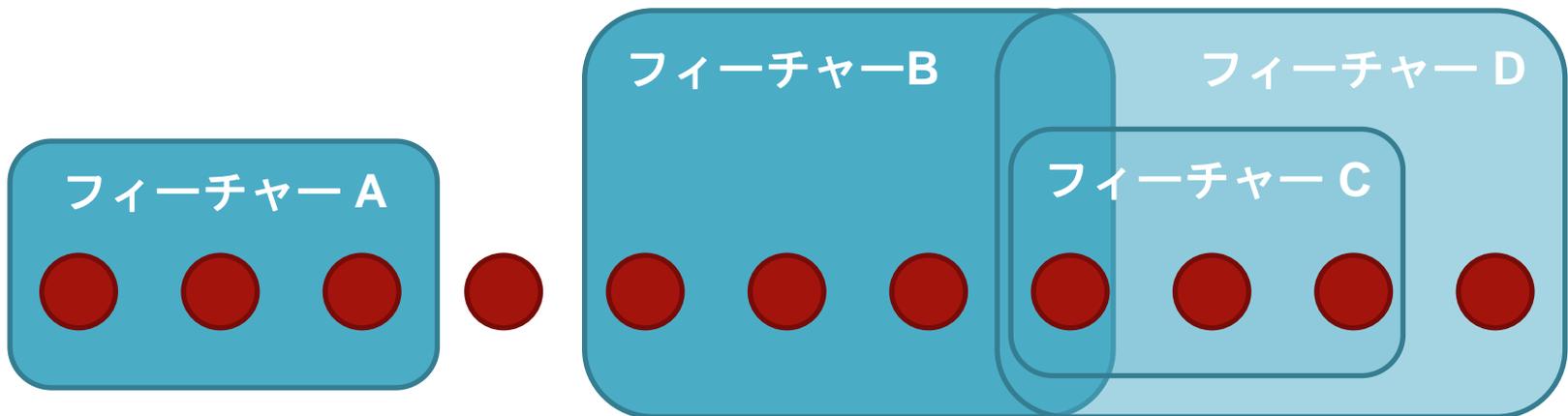
プラグイン

- ❑ EclipseもCSSもプラグインの集合体
- ❑ プラグインの構成を変えることで、様々なCSSを構築することが可能
- ❑ 再利用可能なパーツとして最小の粒度
 - KEK版CSSは100以上のプラグインから成る
- ❑ プラグイン間の依存関係を定義可能



フィーチャー

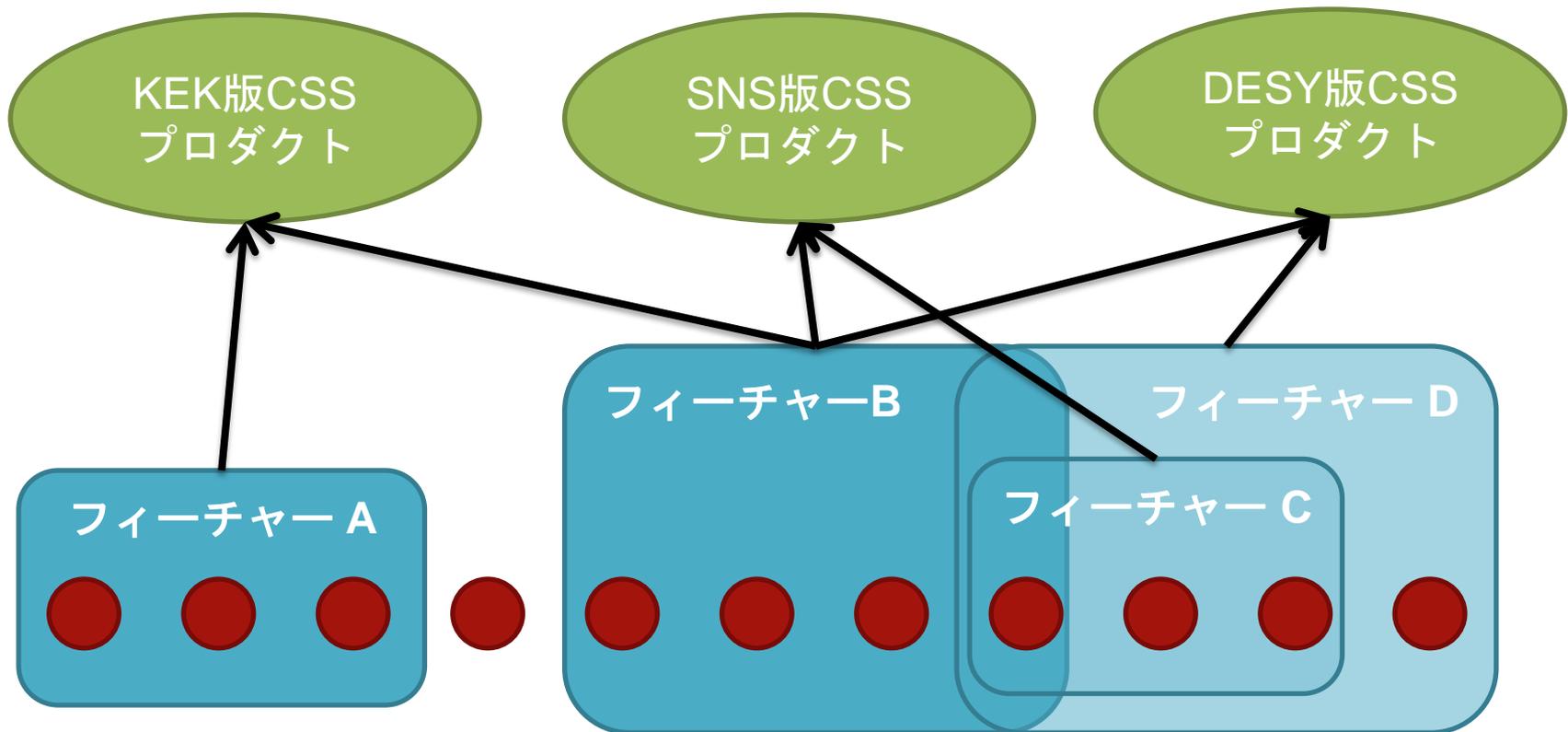
- ❑ プラグインを1つのグループとしてまとめたもの
 - .featureというファイルで定義
 - インストール・バージョンアップはフィーチャー単位
 - 1フィーチャー = 1アプリケーションというイメージ
 - BOY – org.csstudio.opibuilder.feature
 - DataBrowser – org.csstudio.trends.databrowser2.feature
- ❑ フィーチャーは入れ子にすることが可能



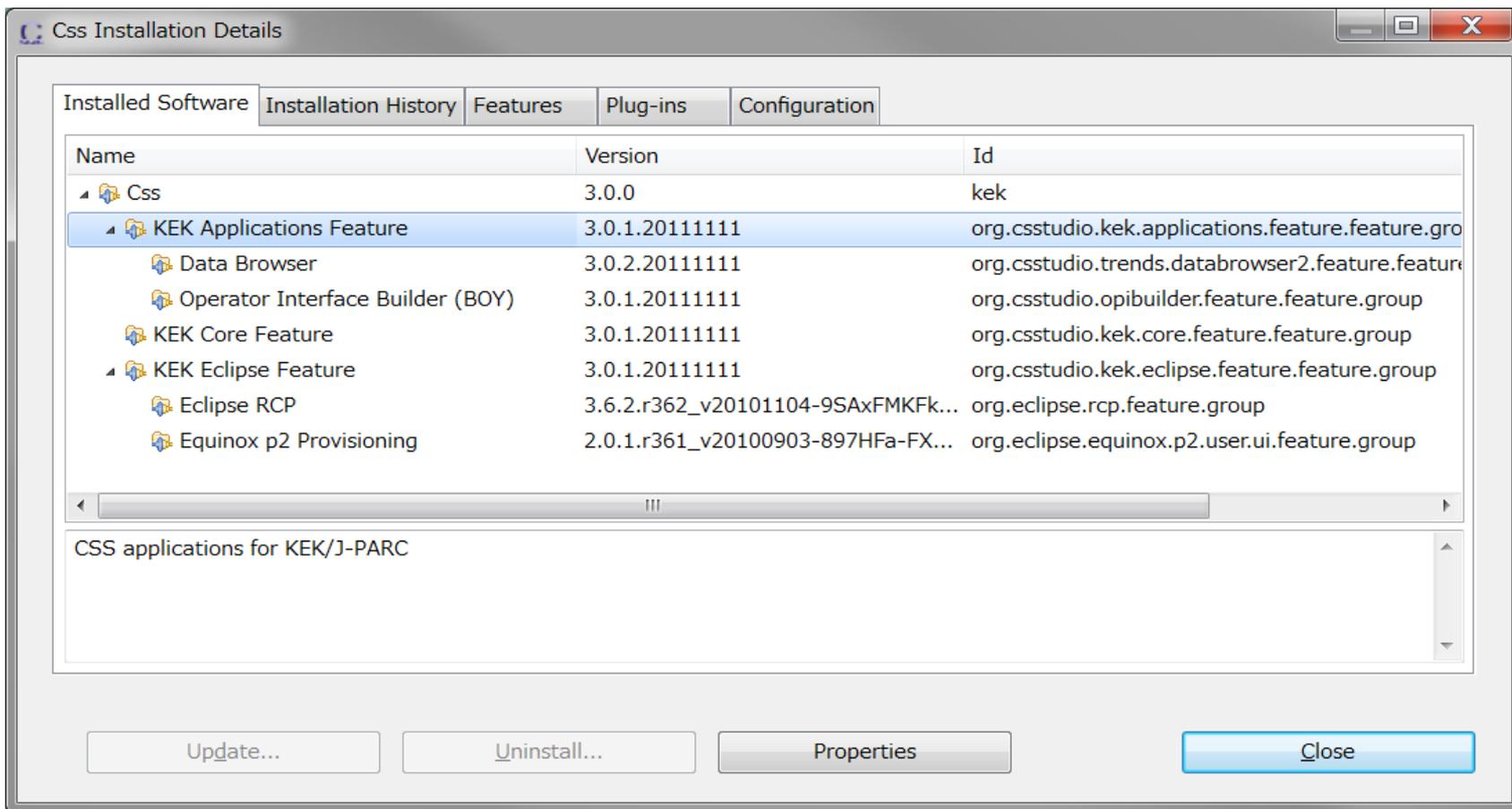
プロダクト

□ プロダクト

- KEK版と呼んでいるものは、一つのプロダクト
- .productファイルで定義



KEK版CSSの場合



□ CSSの拡張性の根幹

□ 例) メニュー

■ Extension Point: org.eclipse.ui.menus

■ 新たにメニューを追加するときには、plugin.xmlのExtensionsでorg.eclipse.ui.menusに対する拡張要素を記述

□ 例) 履歴読み込み

■ Extension Point: org.csstudio.archive.reader.ArchiveReader

■ 新たにkblogの読み込みに対応するときには、このExtension Pointに対して拡張要素を記述

Control System Studio 3.0 プラグイン作成講座

～デモ～

開発環境の設定

開発環境設定の流れ



- ❑ Eclipse RCPのセットアップ
- ❑ CSSのソースコードのダウンロード
 - Mercurialリポジトリから
 - すごく時間がかかる
 - ダウンロード後も重い (バックグラウンドでいろいろ処理)
 - リリースされたバージョンのソースアーカイブから
- ❑ CSSをビルド・実行

Control System Studio 3.0 プラグイン作成講座

～デモ～ プラグインの作成

プラグイン作成の流れ



- プラグインを作成
- フィーチャーの作成 or フィーチャーへの追加
- プロダクトへの追加

Control System Studio 3.0 プラグイン作成講座

～デモ～

PVの取得と画面への描画

デモの流れ



- ビューの作成
- メニューの再構成
- テキストボックスとボタンの作成
- PVへの同期アクセス
- PVのモニタと非同期アクセス

洗練されたアプリケーションへ



□ 学ぶべきこと

- Java
- SWT / JFace

□ 検討すべきこと

- ヘルプ
- ログ
- メニュー
- ツールバー
- ローカライゼーション
- 設定の管理
- パースペクティブ
- アーカイブデータへのアクセス

Control System Studio 3.0 プラグイン作成講座

～デモ～

作成したアプリケーション
の配布

配布方法



- ❑ 自分でビルドしたCSSを配布
- ❑ アーカイブによる配布
- ❑ 更新サイトによる配布
- ❑ Mercurialリポジトリにアップロード (push)
 - SourceForge.net上のMercurialリポジトリにpushするには書き込み権限が必要