

Intrexx Professional Intrexx Compact

RELEASE 5.2



**Einrichten einer Debug-Umgebung in
Eclipse**

Inhaltsverzeichnis


1. Einleitung.....	3
2. Erstellen einer Java-Klasse	3
3. Einbinden von externen Bibliotheken.....	4
4. Einrichten der Debug-Umgebung.....	5
4.1. Einrichten der Konfiguration.....	6
4.2. Die Debug-Perspektive.....	8
4.3. Java-Klasse debuggen	9



Copyright






Das vorliegende Dokument ist in all seinen Teilen urheberrechtlich geschützt. Alle Rechte sind vorbehalten, insbesondere das Recht der Übersetzung, des Vortrags, der Reproduktion und der Vervielfältigung. Ungeachtet der Sorgfalt, die auf die Erstellung von Text, Abbildungen und Programmen verwendet wurde, können weder Autor, Herausgeber oder Übersetzer für mögliche Fehler und deren Folgen eine juristische Verantwortung oder irgendeine Haftung übernehmen.

Die in diesem Werk wiedergegebenen Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. können auch ohne besondere Kennzeichnung Marken sein und als solche den gesetzlichen Bestimmungen unterliegen.



Schreibkonventionen

In diesem Handbuch werden Textstellen *kursiv* dargestellt, wenn sie sich auf Einstellungen in den abgebildeten Dialogen beziehen. Menüpunkte, die in Kontextmenüs erreichbar sind, sind immer auch über das Hauptmenü erreichbar. Hauptmenüpunkte werden nicht beschrieben, es sei denn, sie sind nicht über das Kontextmenü erreichbar. Eine Beschreibung der allgemeinen Hauptmenüpunkte finden Sie im Handbuch  *Portale*. Programmiercode im Text wird in der Schriftart `Courier` dargestellt. Kontextmenüs können mit einem Klick mit der rechten Maustaste auf das beschriebene Element geöffnet werden.

<intrexx> bezeichnet im Folgenden Ihren Intrexx Installationspfad, unter Windows z.B.  `c:\intrexx\`, unter Linux z.B.  `/opt/intrexx/`. Folgende Symbole werden für die Kennzeichnung von speziellen Informationen verwendet:

-  Informationen
-  Verweise auf ein Intrexx Handbuch
-  Verzeichnisse
-  URLs
-  Klick auf Schaltflächen

Vorkenntnisse


Für das Verständnis dieser Dokumentation sind keine speziellen Vorkenntnisse erforderlich. Hilfreiche Informationen finden Sie in den Intrexx Handbüchern  *Setup* und  *Start*.

1. Einleitung

Natürlich ist es für jeden Entwickler auch möglich, eigene Java-Klassen zu erstellen und in ein eigenes entwickeltes Portal zu integrieren. Somit können zu jedem Zeitpunkt eigene Methoden und Funktionen integriert werden und der Umfang eines Portals erweitert werden.

Das Einbinden einer Klasse in ein Portal geschieht mit Hilfe von **Velocity** (siehe: <http://velocity.apache.org/>).

Der Hauptanwendungsbereich liegt im Erstellen von dynamischen Webseiten, kann aber auch in Verbindung mit XML oder gänzlich als alleinstehende Anwendung verwendet werden.

Eine genauere Beschreibung zum Mechanismus von Velocity und zur Benutzung finden Sie im Handbuch  *Einbindung eigener Java-Klassen mit Velocity*.

Im Folgenden wird das Einrichten einer Debug-Umgebung für eigene erstellte Java-Klassen erläutert.



ACHTUNG:

Bei diversen Einstellungen wie Versionsangaben von Intrexx oder Pfadangaben achten Sie bitte genau auf die bei Ihnen gültigen Werte.

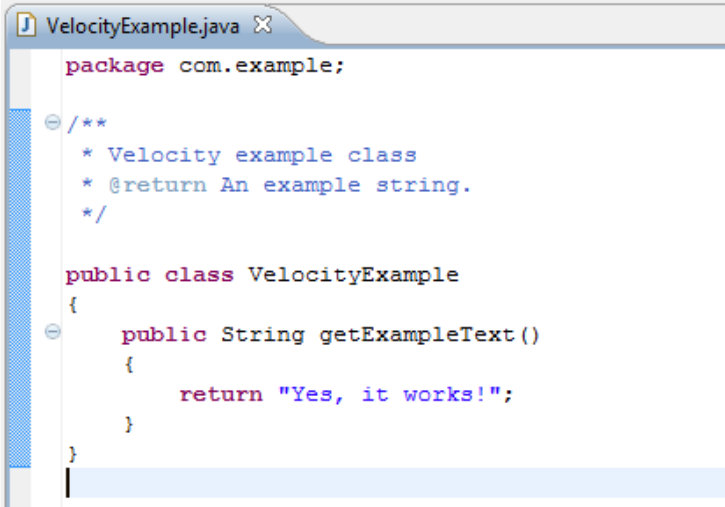
Sollten Sie z.B. statt Version 5.2 Version 5.0 verwenden, achten Sie bitte an den entsprechenden Stellen auf die korrekte Angabe Ihrer Version.

Weiterhin sind die hier verwendeten Klassen- und Projektnamen nicht bindend. Auch hier müssen die entsprechenden Angaben an Ihre Gegebenheiten angepasst werden.

Stellen mit dem Hinweis  **[Copy-Paste]** können direkt aus dem Dokument übernommen werden, um Schreibfehler zu vermeiden.

2. Erstellen einer Java-Klasse

Als Beispiel soll ein String ausgegeben werden, der anzeigt, dass die Einbindung der Java-Klasse in ein bestehendes Portal mittels Velocity funktioniert hat.



```
VelocityExample.java X
package com.example;

/**
 * Velocity example class
 * @return An example string.
 */

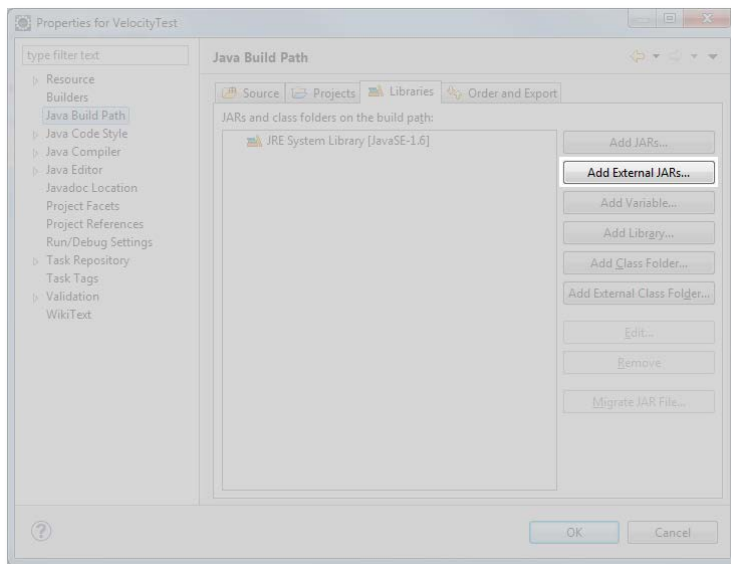
public class VelocityExample
{
    public String getExampleText()
    {
        return "Yes, it works!";
    }
}
```

3. Einbinden von externen Bibliotheken

Im Folgenden müssen die Intrexx-Bibliotheken eingebunden werden.



 Die hier gezeigten Screenshots beziehen sich auf Eclipse Version Indigo.

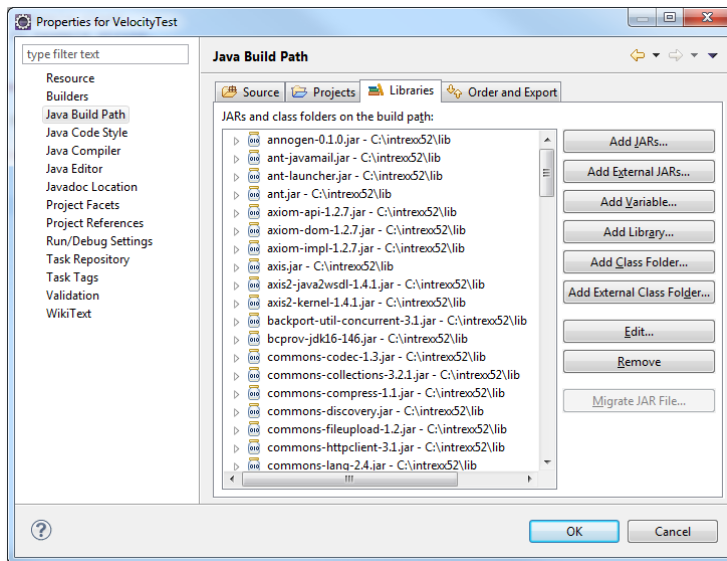
Innerhalb von Eclipse mit dem Mauszeiger das aktuelle Java-Projekt selektieren, das Kontextmenü des Projektordners öffnen und *Properties/Java Build Path / Libraries /Add External Path* wählen.



In dem sich nun öffnenden Dialogfenster muss der Pfad zu den Bibliotheken der installierten Intrexx-Version angegeben werden.

Bitte beachten Sie, dass der Pfad je nach installierter Version und gewähltem Installationsort variieren kann.

Wählen Sie bitte das Verzeichnis  `<intrexx>\lib` und markieren Sie alle Dateien dieses Ordners mit **Strg+A** und klicken Sie dann auf *Öffnen*. Anschließend *Add External Class Folder* auswählen und den Pfad zu  `<intrexx>\lib\update` wählen. Nach dem Hinzufügen sollte die Einbindung in etwa wie folgt aussehen.

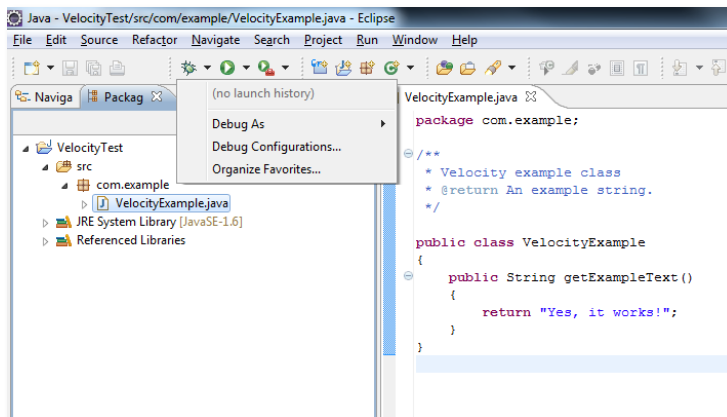


Nun auf den Reiter *Order and Export* wechseln. Klicken Sie auf *Select all* und bestätigen Sie anschließend mit *OK*.

4. Einrichten der Debug-Umgebung

Im Folgenden wird eine komplette Eclipse Debug-Umgebung eingerichtet, die es erlaubt, die erzeugte(n) Klasse(n) zu debuggen.

Klicken Sie auf den Pfeil neben dem grünen Debug-Käfer und im sich öffnenden Menü *Debug Configuration*.



4.1. Einrichten der Konfiguration

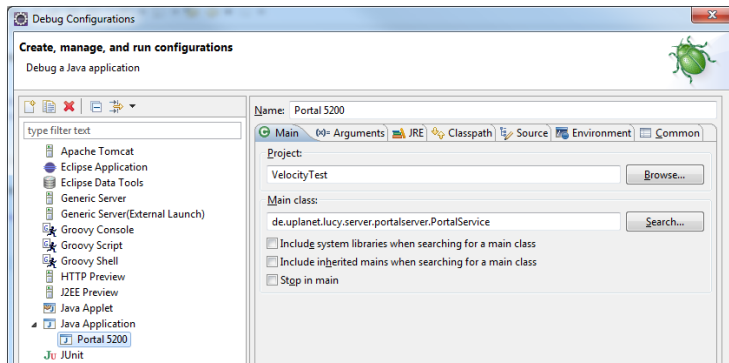
Im Debug-Konfigurationsdialog muss der Menüpunkt *Java Application* gewählt werden. Durch einen Klick auf *New Launch Configuration* erstellen Sie eine neue Konfiguration.

Name:

Frei definierbar. In diesem Beispiel wird *Portal 5200*.

Main class:

de.uplanet.lucy.server.portalserver.PortalService  **[Copy-Paste]**



Wechseln Sie auf den Reiter *Arguments*

VM-Arguments



[Copy-Paste]

-Dfile.encoding=UTF-8

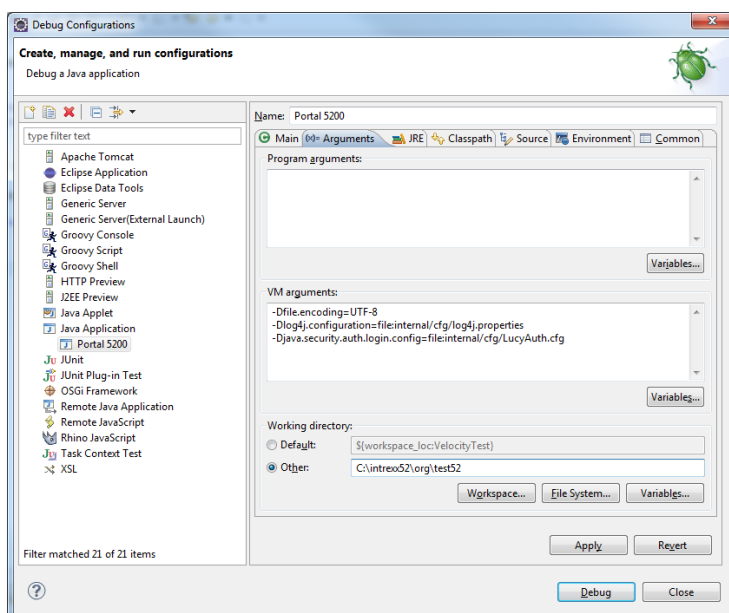
-Dlog4j.configuration=file:internal/cfg/log4j.properties

-Djava.security.auth.login.config=file:internal/cfg/LucyAuth.cfg

Bei *working directory* geben Sie bitte den Pfad zu dem Intrexx Portal an, in dem Sie Ihre Java-Klasse verwenden wollen.


Hierfür klicken Sie auf *File System*, z.B.: <intrexx>\org\test52

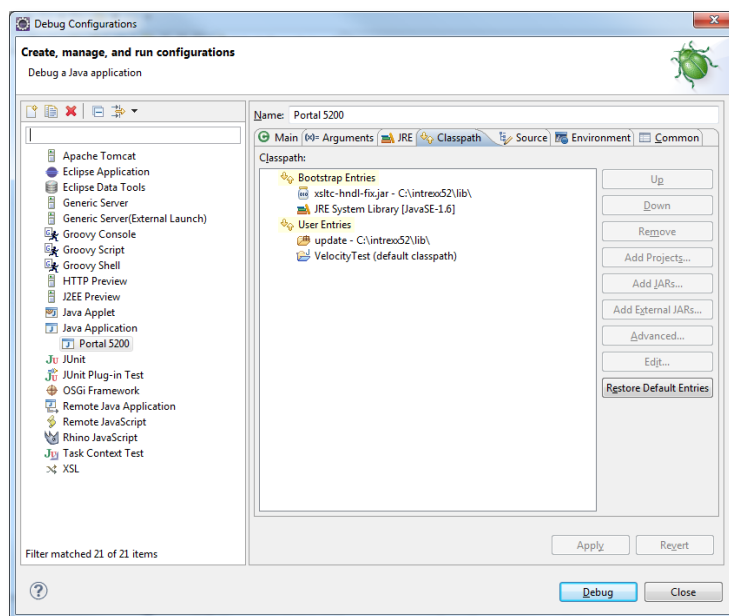
Die korrekten Einstellungen der **portal** Konfiguration.



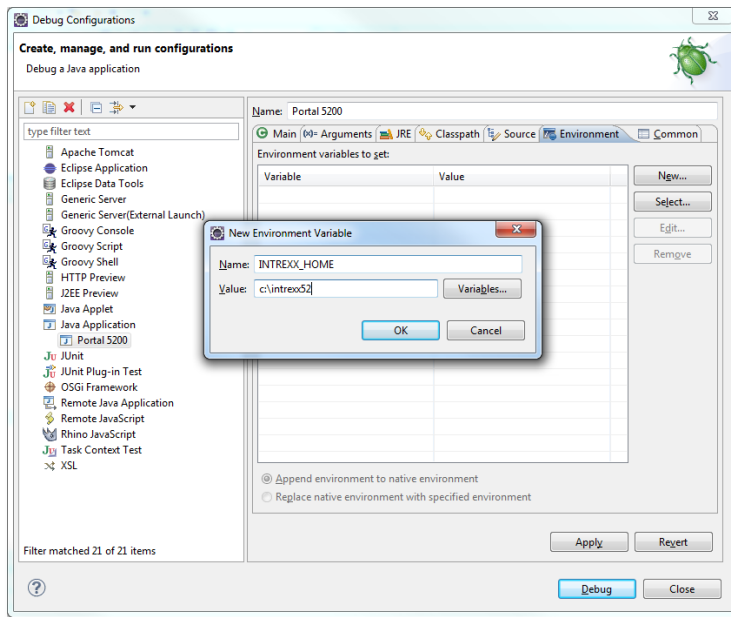
Wechseln Sie nun auf den Reiter *Classpath*. Wählen Sie den Eintrag *Bootstrap Entries* und fügen Sie unter *Add External JARs...* die Datei `<intrexx>\lib\xsltc-hndl-fix.jar` hinzu. Dieser Eintrag **muss** an erster Stelle unterhalb von *Bootstrap Entries* stehen und kann über die Schaltfläche *Up* verschoben werden.

Wählen Sie den Eintrag *User Entries* und fügen Sie über *Advanced...*, *Add External Folder* den Ordner `<intrexx>\lib\update` hinzu und verschieben Sie ihn an die erste Stelle unterhalb von *User Entries*.

Sind all diese Einstellungen korrekt, übernehmen Sie die Änderungen durch einen Klick auf  *Apply*.





Wechseln Sie abschließend auf den Reiter *Environment* und legen Sie über *New...* eine neue Environment Variable mit dem Namen `INTREXX_HOME` und dem Intrexx Installationspfad als Wert an.



In der linken Navigationsleiste sollten sich nun unter dem Menüpunkt *Java Application* die Launch Configuration (z.B. *Portal 5200*) haben.

4.2. Die Debug-Perspektive

 Ab dieser Stelle wird davon ausgegangen, dass Sie bereits die Java-Klasse(n) in Ihr Portal integriert haben. Weitere Informationen über Velocity und über die Einbindung eigener Java-Klassen in Ihr Portal finden Sie im Handbuch  *Einbinden eigener Java-Klassen mit Velocity*.

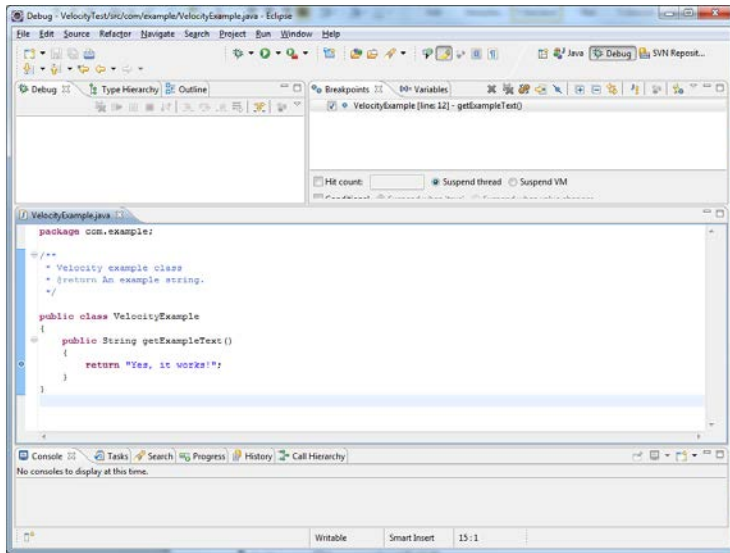
Öffnen Sie über *Window / Open Perspective / Other / Debug* die Debug-Perspektive.

Um abschließend die Debug-Funktionalität zu testen, setzen Sie an einer beliebigen Stelle Ihrer Java-Klasse einen Breakpoint.

Haben Sie das oben genannte Beispiel verwendet, setzen Sie durch Doppelklick auf den grauen Rand links in der Codeansicht einen Breakpoint in der Zeile

```
return "Yes, it works!!";
```


War dies erfolgreich, haben Sie in der gewählten Zeile einen kleinen blauen Punkt. Weiterhin sehen Sie rechts oben in der Übersicht der Breakpoints einen entsprechenden Eintrag (siehe Bild).



4.3. Java-Klasse debuggen

Bevor Sie die Debug-Konfiguration in Eclipse starten, stoppen Sie bitte über die Windows/Linux-Dienstverwaltung oder die Intrexx Service Console den entsprechenden Portaldienst.

Klicken Sie nun auf den Pfeil neben dem Debug-Käfer und starten Sie den Portaldienst im Debug-Modus durch einen Klick auf *5200 portal*.

 Bitte beachten Sie an dieser Stelle nochmals, dass sich die Namensgebung auf das hier gewählte Beispiel und die Intrexx Version bezieht. Besitzen Sie eine andere Version, ergeben sich entsprechend andere Namen für die Prozesse.

Rufen Sie nun Ihr Intrexx Portal, in dem die Klasse verwendet wird, im Browser auf. Sobald Sie die Funktion aufrufen, die den Test-String der Java-Klasse ausgeben soll, wird der Breakpoint erreicht. Sie können dann zurück zur Eclipse Debug-Umgebung wechseln.

Die gerade aktive Zeile wurde markiert und farblich hervorgehoben.

