**Installer**

**for**

**Ignition-based Applications**

**Document Version:**

**Mar 20, 2016**

Table of Contents

Table of Contents

1. Introduction 2

2. Loading 3

3. Creating the Installer Module 4

4. Panels 5

4.1 Welcome Panel 5

4.2 Project Panel 7

4.3 Epilog 7

# Introduction

Ignition™ is an execution and development platform from Inductive Automation designed to support a wide-range of industrial applications. While a basic Ignition-based application may require only a project file, a more comprehensive application may require a number of different components for correct execution. These components may involve a variety of file types and installation steps. The variety may be confusing to the end user. Absence of one or more of these components may yield an incomplete or inconsistent installation and result in subtle (or not so subtle) errors.

This document describes an ILS-Automation product that handles application releases in the familiar paradigm of an installer. Using the ILS Application Installer, the end user is presented with a wizard-style sequence of screens that handle installation of the various components that make up the target application. These components may be, among other things: full or partial projects, global projects, UDT definitions, icons, internal and/or external python packages, SQL update scripts, and Java-based modules.

The release bundle or installer for a particular delivery is packaged into a single file, an Ignition module file. Embedded within the module are all resources required for the target application plus an Ignition project to install it. This project is available only when the install module is loaded into the Gateway. The project supports an end-user wizard-style interface to accomplish the installation.

The screens with which the user interacts are completely configured via an XML file. This file describes both the information on the screens as well as the locations of the actual resources in the bundle. The file is known as the bill-of-materials. The bulk of this document describes this file with sample output.

# Loading

The install bundle is loaded just like any other Ignition module - from the Gateway configuration page. It appears as “ApplicationInstaller” in the module list. This is the default name. It may be easily changed to something more appropriate to the product being installed by editing the application-installer-module.xml file in the module bundle.



Figure – Gateway Modules View

Once the bundle is loaded, a new entry appears on the configuration panel. It names the product and provides a link to execute the installer.



Figure – Entry on Configuration Panel

The name that appears comes from the *<title>* element of the bill of materials.

# Creating the Installer Module

An Ignition module is simply a jar file. It can be created in a number of ways:

* manually – use the *jar* command that is part of any standard java distribution. E.g.

jar –cf <jar-file-name> <directory-to-be-jarred>

* shell script – create a *bash* script to collect artifacts and create the module.
* ant script – incorporate building an installer into your normal build process. The *ant* script may be executed directly from *Eclipse*, for example.

Standard components ,,,

Different than normal in that there is an “artifacts” directory, plus bom.xml.. Locations are relative to the root. Convention places everything under “artifacts”.

Jar command or build directly with ant script.

Sign at Inductive Automation website. You must be registered as an Ignition developer in order to do this.

# Panels

This section describes each panel type in detail. Not every panel is applicable for every product. In some cases multiple versions of the same panel may be needed. This is all controlled by the makeup of the XML file, the product bill-of-materials.

Panel type is fixed to a fixed vocabulary. Each type correlates to a separate Java class that renders the page. Within a bill-of-materials, the combination of panel type and subtype must be unique.

## Welcome Panel

The initial panel displays a description of the product being installed along with a comparison list of the current installed state. It also provides an opportunity to view release notes.

An initial section of the XML document contains a list of properties that describe the new release of the product. On startup, these are displayed on the Welcome panel along with corresponding values describing what is currently installed. These values are updated in the internal Gateway database once the installation is complete.

The properties shown in the XML fragment below should be included in every update. These are:

* product – the name of the product
* release – marketing name of this update
* date – release date
* version – an integer release number that increments with each update

These same properties are defined in the bill-of-materials as shown below:

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<installer>

<title>ACE Controls Installer</title>

<description>Installer for Ignition Application.</description>

<property name=*"product"*>ACE Controls</property>

<property name=*"release"*>1.2b5</property>

<property name=*"date"*>July 4, 2016</property>

<property name=*"version"*>2</property>

The welcome panel is defined as:

<!-- The welcome panel must always be displayed, thus no version -->

<panel type=*"welcome"* essential=*"True"*>

<title>ILS Automation Installer for ACE Controls Applications</title>

<preamble>Welcome to the ILS Automation installer for the ACE Controls

Ignition Application. The table below compares the existing

installation of this product, if any, with the versions

contained in this new installation. Before continuing,

please make sure that you have defined tag providers and database connections for both production and isolation

(test) environments. Also make sure that you have write

permission to the "lib" folder within the Ignition

installation directory.

</preamble>

<artifact name=*"notes"* type=*"pdf"*>

<location>artifacts/doc/ReleaseNotes.pdf</location>

</artifact>

</panel>

The *<artifact>* element defines where the resource is located within the installation module.



Figure – Welcome Panel

The panel has a button for displaying release notes, and checkboxes that provide filters for panels to be displayed. Options include the ability to skip artifacts that have not been updated in the current release. Additionally a choice may be made to skip artifacts that are not necessary for production, such as documentation, test scripts or source code.

## Project Panel

Blah.

The artifact subtype attributes control the creation and update behavior for the selected project. Valid subtypes are:

* full – the internal artifact is installed as a new project, completely replacing any existing project with the same name.
* global – the internal artifact is a partial project which will be merged with the current global project. The internal project’s resources will overwrite any resources that exist both projects
* partial – the internal artifact is a partial project which will be merged with the selected existing project. The internal project’s resources will overwrite any resources that exist both projects.

## Epilog

A final panel is shown.