pCon.planner Plugin SDK

Version 8.16

Getting Started with the pCon.planner Plugin SDK

Welcome to the pCon.planner Plugin Software Development Kit. We appreciate your interest in interacting with the pCon.planner application.

Content

1 SDK Content....2
2 Development....2
3 Registration....3
4 Installation....4
5 Loading....5
6 Distribution....5
7 Support....5
8 Plugin registration....5
9 Integration....5
    9.1 Plugin structure....5
    9.2 Plugin description....6
10 Disclaimer....6

Copyright EasternGraphics GmbH

1 SDK Content

The SDK contains the following items:

- `\Documentation`
  - Generated API documentation in PDF and HTML form
  - XML Schema for the Plugin Info
- `\SamplePlugin`
  - A Visual Studio sample plugin project
- `\SamplePlugin\X3gPlugin`
  - `X3gPlugin.dll` - .net assembly to be included as reference in your plugin project
  - `X3gPlugin.xml` – Support for Visual Studio auto-completion

  NOTE: The files are copies from the pCon.planner \bin folder. These are just included here to make the SDK self-contained. The latest version of these files is always located in current pCon.planner distribution. Your project should use the files from there!
- `\Plugin Binaries`
  - Output of compiled SamplePlugin – ready to run in pCon.planner

2 Development

Plugin development for pCon.planner is based on the Microsoft .NET platform. We rely on .NET Framework 4.8.1 which should be installed on all supported platforms.

Copyright EasternGraphics GmbH

3 Registration

A plugin need a valid hash in order to be loaded (see plugin XML: `hash` attribute). This hash will be provides by EasternGraphics after registering the plugin.[1]

Before you register you plugin you can play with the integrated SamplePlugin, it contains a valid hash.
If you like to create your own projects you can ask for a special time limited developer license[2]. That license allows running plugins without registration. In this case the hash has to be "0". These plugins will no run outside the machine containing the developer license.

To register your plugin, please send a request to request@easterngraphics.com, including a brief description of your plugin and provide your vendor (`X3gGetVendor()`) and plugin name (`X3gGetName()`).

1 A special trial license allows working without valid hash for a specific amount of time on a dedicated development system.
2 Ask your contact or licensing@EasternGraphics.com for a "pCon.planner Plugin SDK" developer license.

Copyright EasternGraphics GmbH

4 Installation

To install a plugin it has to be in one of the standard folders or in any registered folder. Plugins are looked-up in the following order:

➔ *Application sub-folder*

`<programs>\EasternGraphics\pCon.planner\plugins`

Use only for development and testing!

➔ *Registered folders*

To register a folder it has be listed in the Windows Registry in one of the following locations:

`HKEY_LOCAL_MACHINE\SOFTWARE\EasternGraphics\X3GPlugins`

`HKEY_CURRENT_USER\SOFTWARE\EasternGraphics\X3GPlugins`

A path has to been added there as a new string value with any unique key (GUID or plugin name with version suffix), e.g:

`"{C197EE96-9B28-473D-AF81-EAA431CEC15F}"=<Path>`

`"MyUniquePluginName"=<Path>`

The registered path is the parent folder of the plugin folder.

➔ Standard folders

- All Users folder:
  ➔ `<ProgramData>\EasternGraphics\<APP>\plugins`
- User folder:
  ➔ `<USER>\AppData\Roaming\EasternGraphics\<APP>\plugins`

`<APP>` has to be "pCon.planner Pro" or (more generic) "pCon.planner".

Copyright EasternGraphics GmbH

5 Loading

In case of duplicate plugin registration the plugins with the highest versions are tried to load first – if loading fails the next lower version is used.

Currently all plug-ins are automatically loaded on pCon.planner start-up. A plug-in must not do any significant processing when it is loaded. All initialization in the `X3gInitialize()` method should be as fast as possible. Later version might add a timeout. Loading time below 1s on average system should be fine.

6 Distribution

A plugin does not have to rely on dependency which are fulfilled by pCon.planner (e.g. VC runtime, sqlite3.dll). The installed components might change in future and we will not take care of backward compatibility.

7 Support

Please note that we do not offer free support for this product. Currently, there is also no dedicated online forum or mailing list available. If you require technical assistance, please contact us via the following email address only: support@easterngraphics.com

8 Plugin registration

To be accepted by the application a plugin requires a fingerprint. This fingerprint is a hash value which has to be provided by the plugin description XML and by the plugin API (`X3gGetGlobalIdentifier()`). See 3 how to get a valid fingerprint.

9 Integration

9.1 Plugin structure

A plugin consists of at least two files: A .net based DLL and a XML-based plugin description. The files must reside in a pCon.planner plugin folder within a sub-folder of the plugin name.

Example for a plugin called "`SamplePlugin`":

```
..\pCon.planner\plugins\SamplePlugin\
SamplePlugin.dll
SamplePlugin.xml
SamplePlugin.ui (optional)
```

Copyright EasternGraphics GmbH

9.2 Plugin description

The plugin description `(SamplePlugin.xml)` has to be conform with `PluginInfo.xsd` (see `Documentation\` folder). It describes the dependencies of the plugin and contains a fingerprint to validate the plugin.

Dependencies can by on version-ed modules. Either specific files (`filename` attribut) or symbolic modules (`name` attribut). The following symbolic modules are defined:

- `core` – reefers the kernel version
- `planner` – reefers to pCon.planner as target application

10 Disclaimer

EasternGraphics reserves the right to change the interfaces described at any time to any extent. We try to keep the interfaces as compatible as possible with each update, but we cannot guarantee that. All conscious changes are documented to the best of our knowledge.

Contents of the interface that are not explicitly documented can be changed without notice. This particularly applies to content that can be queried dynamically (e.g. keys of properties or properties itself).

Copyright EasternGraphics GmbH