PiSiDo Hilfe

Paketbau Basics

Ein Pisi Paket kann gebaut werden mit dem Befehl “pisi build” command. Der Pisi Baubefehl benötigt eine pspec.xml als argument, und es produziert eine binäre Datei zum installieren. Es ist aber nicht nur alleine die pspec.xml Datei, sondern es benötigt auch noch andere Dateien um ein Pisi Paket zu bauen. Hier ein einfaches Layout der Bau Dateien die benötigt werden :

```
<package_name>
  |__ actions.py
  |__ pspec.xml
  |__ translations.xml
  |__ files
  |__ comar
     |__ packages.py
     |__ service.py
```

<package_name>, Datei und comar sind Verzeichnisse. Lasssen Sie uns einen Blick darauf werfen:

<package_name> : Hier ist ein Paket Name erforderlich, es ist die Bezeichnung für ein Verzeichnis das erstellt wird, für das zu bauende Paket.

actions.py : Diese Datei definiert die configure, make und make install Schritte. Es benutzt die Actions API um zu definieren was mit jedem Schritt zu tun ist. PiSiDo hat einige Vorlagen für Sie zur verfügung sowie autotools, cmake, kde4 kde5, qt4, qt5, etc. Für mehr Informationen besuchen Sie die Pardus developer pages.

pspec.xml : Diese Datei beinhaltet Paket Informationen. Es beinhaltet die Paket Quelle, detaillierte Informationen über das Paket und die history der build files. Die Datei hat eine RNG rule file. Sie können es finden hier. Für mehr Informationen besuchen Sie die Pardus developer pages.

translations.xml : Diese Datei ist optional. Diese Datei beinhaltet eine zusammenfassung der Paket Beschreibung und die sprachliche Übersetzung.

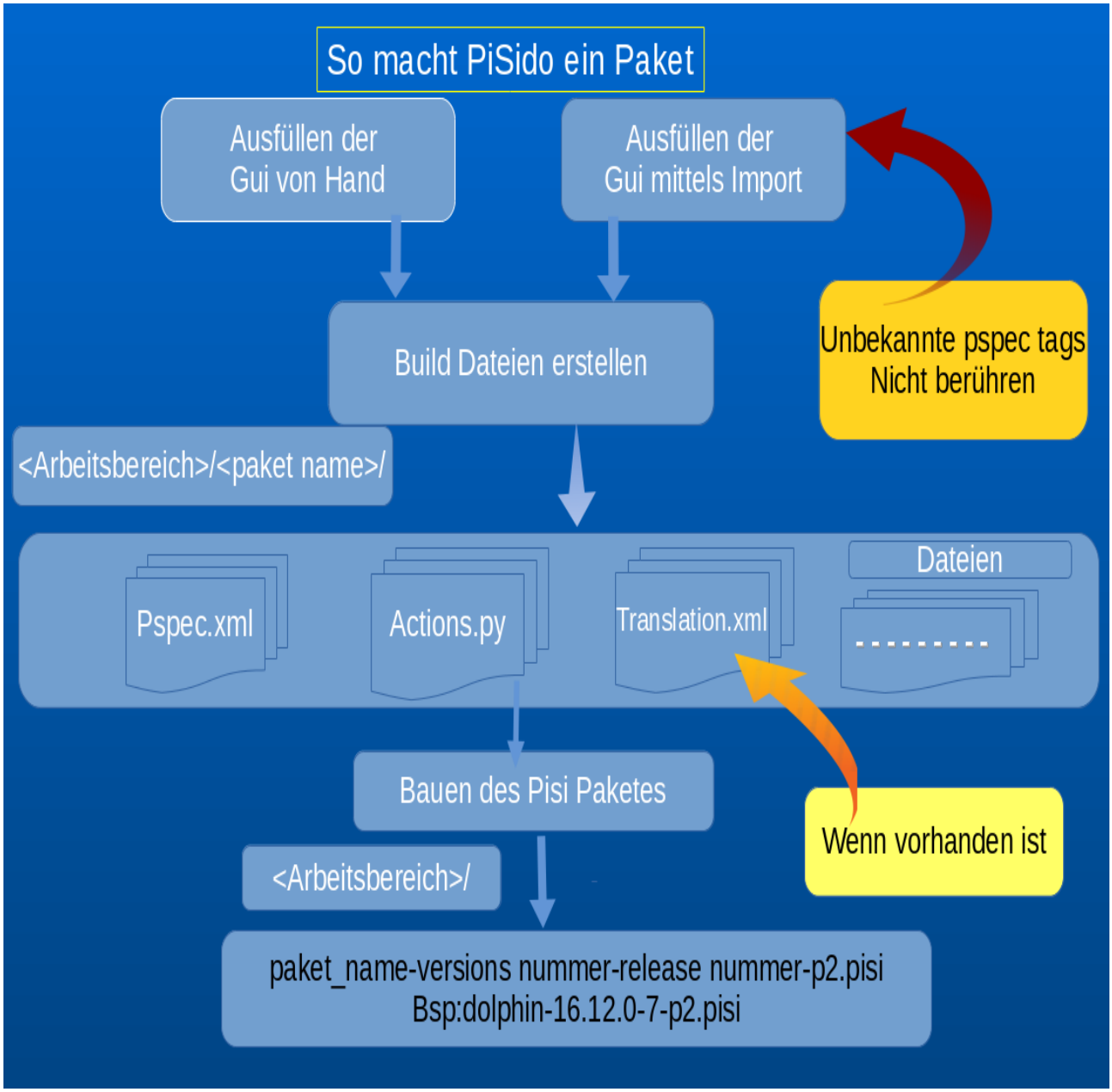
files : Dieses Verzeichnis ist optional, Sie können hier Dateien hinzufügen die im Quellpaket nicht vorhanden sind. Sie können also, patches in diesem Verzeichnis hinzufügen. Jede der hier hinzugefügten Datei muss auch in der pspec.xml auftauchen so das die Dateien beim Bauen miteingebracht werden. PiSiDo hat hierfür grafische tools zur verfügung um Ihnen die Arbeit so leicht wie möglich zu machen.

comar : Dieses Verzeichnis ist auch optional. Diese Verzeichnis beinhaltet COMAR (COnfiguration MAnageR) dateien. Diese Dateien helfen Pisi Linux service operationen zu erstellen. Dieses liegt ausserhalb des PiSiDo's bereiches, und PiSiDo dieses Verzeichnis und solche Dateien nicht. Für mehr Informationen besuchen Sie Pardus developer pages.

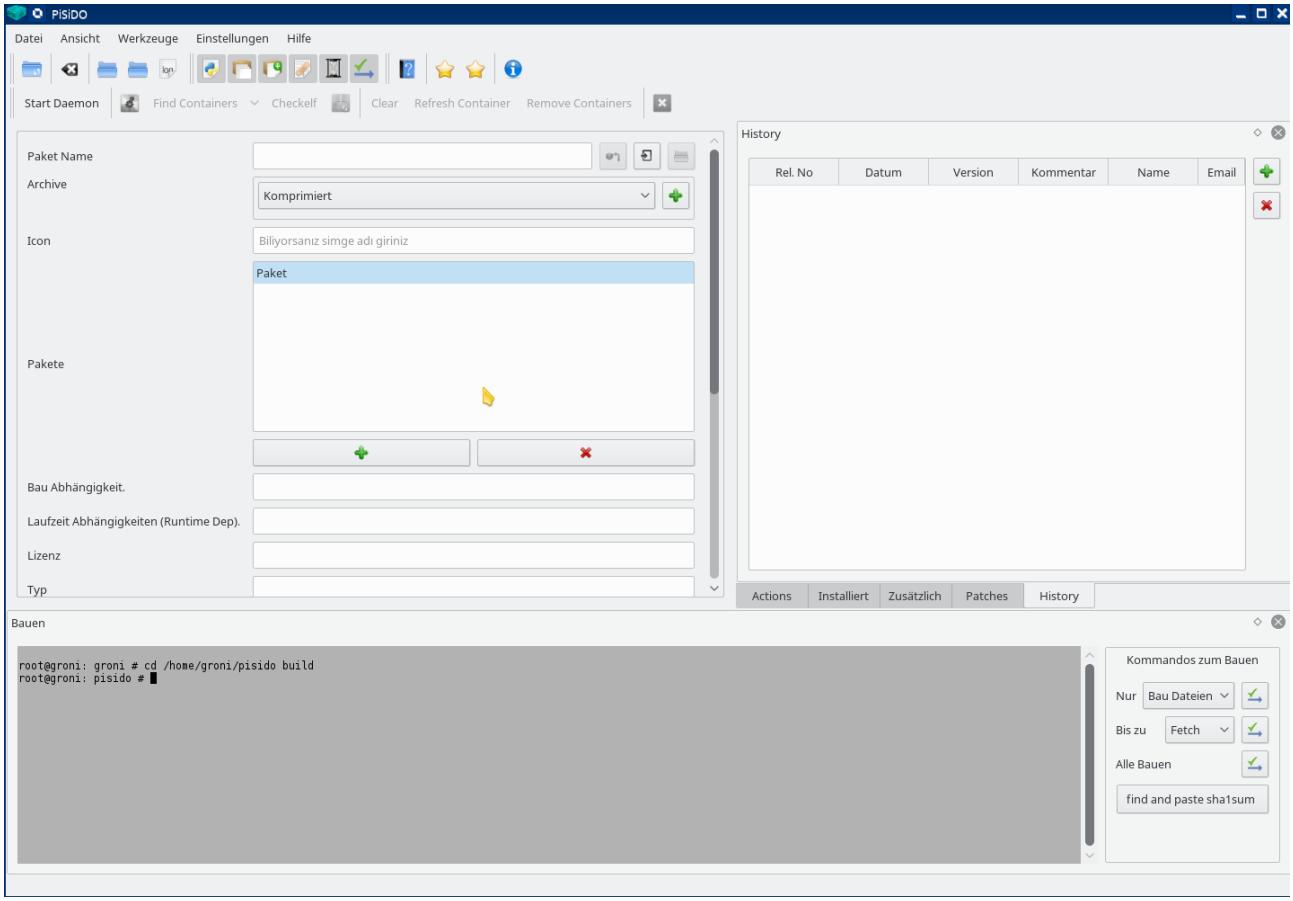
Um ein Pisi Paket zu erstellen benötigen Sie als erstes eine actions.py und eine pspec.xml . Danach können Sie mit dem folgenden Befehl bauen :

```
$ sudo pisi build pspec.xml
```

Sie können wichtige Informationen auch im offiziellen Pardus Wiki nachlesen. Weitere Dokumente finden Sie hier Pardus developer pages.



PiSiDo GUI



Haupt Felder

Paket Name:

Name des Pakets. Dieses Feld wird verwendet, um ein Verzeichnis zu erstellen, wie es unter dem gleichen Arbeitsbereich benannt wird. Auf jedem Zeichen, das Sie eingeben, wird die Anwendung prüfen, ob in den vorhandenen Paket und Verzeichnisse und Listen erstellt werden müssen. Und Sie können ein vorhandenes Paket importieren, indem Sie die Eingabetaste drücken.

Archiv:

Hier, können Sie das Quell Paket definieren. Die Auswahlbox bietet Ihnen zwei Möglichkeiten:

1. Komprimiert: Für locale Dateien. Die Anwendung liest die sha1 Prüfsumme vom Quellpaket automatisch ein, nachdem Sie es hinzugefügt haben.
2. Url: Für lokale und Pakete aus dem Internet müssen Sie den Archiv Pfad und die sha1 Prüfsumme von Hand eingeben.

Tipp : Haben Sie eine falsche sha1 Prüfsumme und versuchen zu bauen. Pisi lädt das Quell Paket ins Archive Verzeichnis. Danach können Sie ein Terminal öffnen um die sha1 Prüfsumme zu bekommen. Enter 1111....11 und bauen. Sollten Sie eine Fehlermeldung bekommen, versuchen Sie:

```
$ sha1sum /var/cache/pisi/archives/pisido-2.0a.tar.gz
```

Build Abhängigkeiten.:

Hier können Sie die Bau Abhängigkeiten definieren. Das Pisi Build System installiert diese Pakete erst, bevor der eigentliche Bau-Prozess startet. Sie können mehr als eine Abhängigkeit eintragen, in dem Sie die Namen mit einem Komma trennnen. Außerdem können Sie die Version einschränken, indem Sie diese speziellen Parameter schreiben:

- qt-devel[>4.7] : qt paket version muss größer sein als 4.7
- qt-devel[=4.7] : qt-devel Paketversion muss gleich sein 4.7
- qt-devel[<4.7] : qt-devel Paketversion muss kleiner sein als 4.7
- qt-devel[>>4.7] : qt-devel Paket muss größer sein als 4.7
- qt-devel[==4.7] : qt-devel Paketfreigabe muss gleich sein 4.7
- qt-devel[<<4.7] : qt-devel Paketfreigabe muss kleiner sein als 4.7

Als Beispiel: qt-devel[>4.7], gtk2-devel[2.20]

Runtime Abhängigkeiten.:

Hier können Sie Laufzeitabhängigkeitspakete definieren. Pisi wird Sie vor der Installation des Pakets installieren. Sie können mehr als eine Abhängigkeit schreiben, die jeweils durch ein Komma getrennt sind. Außerdem können Sie die Version einschränken, indem Sie diese speziellen Parameter schreiben:

- qt[>4.7] : qt Paketversion muss größer sein als 4.7
- qt[=4.7] : qt Paketversion muss gleich sein als 4.7
- qt[<4.7] : qt Paketversion muss kleiner sein als 4.7
- qt[>>4.7] : qt Paket Freigabe muss größer sein als 4.7
- qt[==4.7] : qt Paketfreigabe muss gleich sein 4.7
- qt[<<4.7] : qt Paketfreigabe muss kleiner sein als 4.7

Als Beispiel: qt[>4.7], gtk2[2.20]

Lizenz:

Hier schreiben Sie bitte die Lizenz des Pakets rein.

IsA:

Hier können Sie den Pakettyp auswählen. Wie Anwendung, Service, Lokalisierung, etc.

PartOf:

Hier können Sie die Komponente des Pakets definieren. Wie Desktop, Spiel, Hardware, etc.

Homepage:

Hier bitte die Homepage des Pakets eintragen.

Summary:

Hier können Sie eine kurze Beschreibung festlegen. Dies wird dem Benutzer angezeigt, während er die Pakete auflistet.

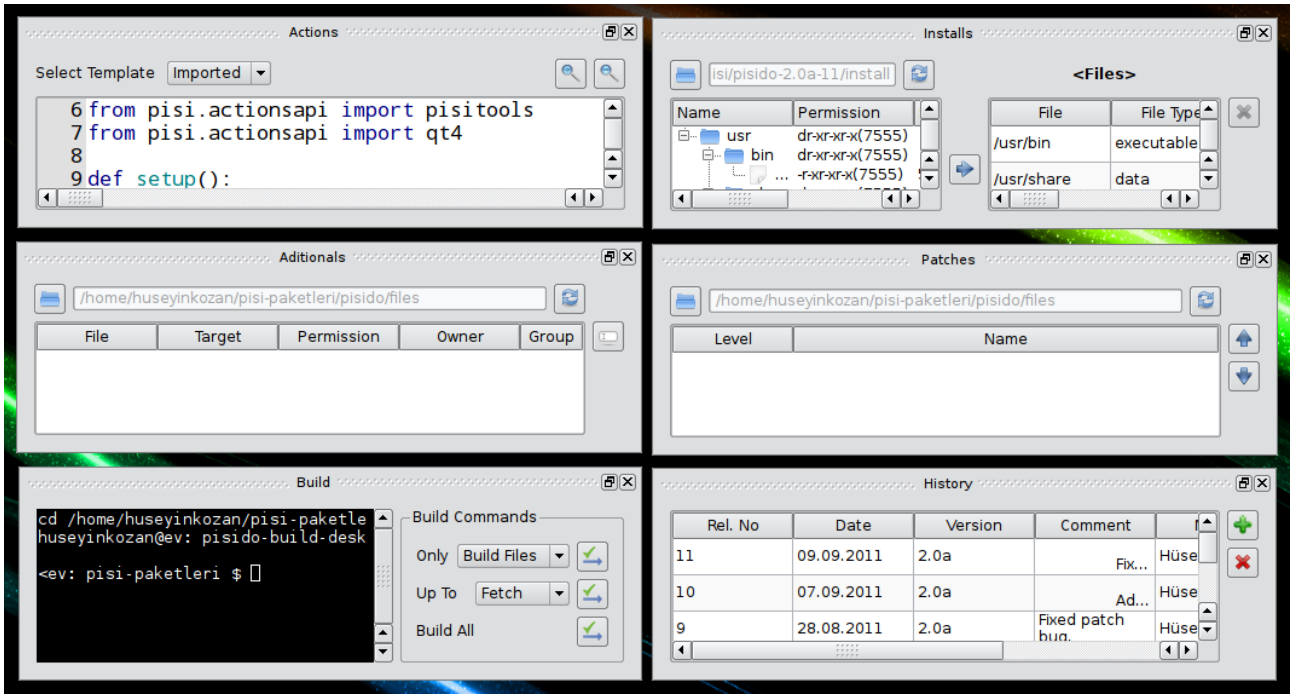
Description:

Hier können Sie eine detaillierte Beschreibung des Pakets einfügen.

Language:

Hier können Sie Zusammenfassung und Beschreibung für andere Sprachen definieren. Beim Importieren werden nicht vorhandene Summary-Description-Paare und falsche Tags nicht geladen.

Dock Fenster



Actions File

Actions-Datei wird für die Verwaltung des Build-Prozesses vom pisi build-System verwendet. Sie können Vorlagen in dem Auswahlfeld ändern. Actions-Datei verwendet python und Actions API. Weitere Informationen über die ACTIONS API finden Sie unter Pardus developer pages.

Actions-Datei Vorlagen haben einige Kommentare, um den Benutzer zu helfen. Außerdem gibt es einige spezielle Variablen, die PiSiDo spezifisch sind. Diese Variablen ersetzen die richtigen Werte beim Erstellen der Build-Dateien. Spezielle Variablen sind :

- __package_name__ : replaces Package Name field
- __version__ : replaces last update release number from History Window
- __summary__ : replaces Summary field

Warning ! : Ihre Änderungen werden nicht gespeichert, bis Sie die Build-Datei erstellen. Die Erstellung der Build-Datei erfolgt automatisch durch alle Befehle, in dem Build Fenster.

Install Files

Hier können Sie festlegen, wo sie jede installierbare Datei installieren. Die Installationsliste wird nach einem erfolgreichen Build abgelegt. Die Anwendung überwacht die Pakete im pisi build Verzeichnis wegen der Änderungen. Wenn Sie ein Build starten und es fehlschlägt, wird die Dateiliste gelöscht.

Als Pflichtfeld gibt es ein Standard-Label, das das Root (/) des Dateisystems für alle installierenden Dateien im Paket definiert. Sie sollten die richtigen Speicherorte für die Dateien oder Ordner hinzufügen und die Standardbeschriftung löschen.

Aditional Dateien

Dies ist ein optionales Feld. Die Anwendung überwacht die Datei für Änderungen unter <Arbeitsbereich> / <Paketname> / Dateien / Verzeichnis. Sie können manuell neue Dateien zu diesem Verzeichnis hinzufügen und definieren, wo es während der Installation in diesem Fenster geht.

Patches

Dies ist ein optionales Feld. Die Anwendung überwacht die Datei für Änderungen unter <Arbeitsbereich> / <Paketname> / Dateien / Verzeichnis. Sie können manuell neue Patches (* .patch) zu diesem Verzeichnis hinzufügen und die Patchreihenfolge in diesem Fenster definieren.

History

Diese Felder zeigen den Verlauf der Paket-Build-Dateien an. Er wird in der Datei pspec.xml vom Paketersteller definiert. Sie können neue Aktualisierungsinformationen hinzufügen. Beachten Sie auch, dass Sie Paketersteller-Informationen definieren können Application Configuration.

Build

In diesem Fenster können Sie einige Befehle eingeben, um Ihr Paket zu erstellen. Hier ist die Liste der unterstützten Befehle :

- Only
  - Build Files : Erstellt die Build-Dateien und stoppt.
  - Package : Erstellt die Build-Dateien und versucht, einen erfolgreichen Build zu machen. Dies ist eine nützliche Anwendung um Änderung in der actions.py zu machen und es ist nicht Notwendig, das Paket neu zu kompilieren. Dies kann als ein letzter Schritt von bis zu sein.
- Up To
  - Fetch : Erstellt die Build-Dateien und gibt pisi build pspec.xml -fetch Befehl. Das wird Quell-Archive herunterladen und stoppt.
  - Unpack : Fetch-Schritt anwenden und sha1-Prüfsumme des Quell-Archivs überprüfen, entpacken, Patches anwenden und beenden.
  - Setup : Anwenden Auspacken Schritt und Anwendung Setup (configure) Schritt und Stoppt.
  - Build : Wendet den Setup Schritt an und übernimmt den Build (kompilieren) Schritt und Stoppt.
  - Install : Wendet den Build Schritt an und installiert und stoppt.
  - Check : Anwenden und Installieren Schritt und prüft Paket und stoppt.

Für mehr informationen besuchen Sie die Pardus developer pages.

PiSiDo Menüs

Datei

Ändern Workspace

Ändert den Arbeitsbereich. Arbeitsbereich ist das Hauptverzeichnis, die Anwendung speichert die Paket-Build-Dateien und gebaute pisi-Dateien. Sie können das Öffnen des Arbeitsbereich-Dialogfelds von sich aus aktivieren oder deaktivieren.

Zurücksetzen

Löscht alle grafischen Schnittstellenfelder.

Exit

Beendet das Programm. Die Anwendung warnt Sie nicht vor nicht gespeicherten Änderungen.

Ansicht

Sie können jedes Dockfenster und jede Symbolleiste in diesem Menü anzeigen und ausblenden.

Werkzeuge

Öffnen Workspace

Öffnet den Arbeitsbereich, den Sie wählen, während Sie die Anwendung mit einem Dateimanager starten.

Öffnen PISI Packaging Directory

Öffnet das pisi-Arbeitsverzeichnis, das sich unter / var / pisi mit einem Dateimanager befindet. Diesen Pfad können Sie ändern in Einstellungen.

Einstellungen

Sprache

Hier können Sie die Standardsprache ändern. Die Änderung wird nach dem Neustart der Anwendung wirksam.

Konfigurieren

Sie können hier anwendungsspezifische Einstellungen ändern. Sie können Packager-Informationen definieren, die Ihnen beim Hinzufügen eines Updates zum Verlauf helfen. Sie können pisi Verpackungsverzeichnis und ändern die Actions API seite und PISI Spec Datei Pfad.

Hilfe

Hilfe...

Öffnet die richtige Hilfedatei über die ausgewählte Sprache.

PISI Spec

Öffnet PISI Spec RNG-Datei. Dieser Wert kann geändert werden in der Einstellungs Konfiguration.

Actions API

Öffnet die ACTIONS API-Seite. Dieser Wert kann geändert werden in der Einstellungs Konfiguration.

Über...

Zeigt Anwendungsinformationen an.

Über Qt...

Zeigt Qt-Toolkit-Informationen an.