Gem #105: Леди Ada целует питона (Python) - Часть 1

Автор: Emmanuel Briot, AdaCore

Эта серия из двух Gems объясняет, как использовать коллекцию компонентов GNAT для взаимодействия вашего кода Ada с Python. Первый Gem описывает преимущества, которые это может принести вашему приложению, а также трудности, возникающие при непосредственном взаимодействии с библиотекой Python. Второй Gem покажет, как начать работу с GNATCOLL, чтобы значительно упростить процесс взаимодействия.

Давайте начнём…

Коллекция GNAT Components (GNATCOLL) с самого начала включала коллекцию пакетов, позволяющую легко взаимодействовать с вашими приложениями Ada со сценариями. Это слой, используемый в IDE GPS для обеспечения расширяемости через оболочку GPS или Python. Использование с оболочкой GPS - это просто игрушка, которую мы первоначально использовали для загрузки процесса, и была сохранена только для обратной совместимости. С другой стороны, Python - это широко используемый объектно-ориентированный язык, который поставляется со своей собственной библиотекой времени выполнения и может быть легко расширен в C или Ada.

Этот Gem не будет вдаваться в подробности самого Python. В Интернете есть отличные учебные пособия. Вместо этого мы сосредоточимся на преимуществах, предоставляемых GNATCOLL над прямым интерфейсом с помощью импорта прагмы и покажем, как сделать приложение сценарием в Python.

Пакет, предоставляющий эту поддержку, называется GNATCOLL.Scripts. Как следует из его названия, он должен быть API-интерфейсом для сценариев-агностиков. Это означает, что ваше приложение ничего не знает о Python или его API. Вместо этого вы экспортируете некоторые классы и функции из Ada в GNATCOLL. Затем последний будет следить за тем, чтобы эти функции были доступны для всех поддерживаемых языков. Хотя в настоящее время мы поддерживаем только Python и оболочку GPS, в будущем могут быть добавлены дополнительные языки (например, JavaScript, Lisp или Caml), и ваше приложение автоматически будет доступно через скрипты.

GNATCOLL также выполняет большую часть управления памятью от вашего имени. Например, Python использует подсчёт ссылок для обнаружения, когда объект может быть освобождён. В реализациях Lisp, таких как Scheme, обычно используется форма сбора мусора. Но это детали, о которых вам не нужно знать, когда вы программируете через GNATCOLL.Scripts. В этом пакете управляются типы Ada (и сами используют подсчет ссылок, как мы подробно описали в более раннем Gem), и автоматически освобождают память, когда больше не используются. Это, конечно, меньше подвержено ошибкам и позволит избежать большого количества утечек памяти в вашем приложении.

Фактически, GNATCOLL также предоставляет несколько небольших расширений для скриптового пакета, если вы также программируете графический интерфейс на основе GtkAda. Последняя (или, скорее, базовая библиотека gtk +) также использует собственный подсчёт ссылок. В результате все может стать действительно сложным, если у вас есть объект Ada, который экспортируется на Python, и этот объект связан с одним из элементов GUI вашего приложения. Выяснить, когда память безопасна для освобождения, требует большой осторожности, но GNATCOLL заботится обо всем этом от вашего имени!

Каковы преимущества взаимодействия с Python? Python, как и многие языки сценариев, имеет довольно высокоуровневый API программирования. В частности, это делает создание сложных структур данных относительно легким, так как оно обеспечивает важный набор таких структур данных, которые полностью интегрированы в язык, а также благодаря его способности к интроспекции и отражению. Наш опыт работы с GPS заключается в том, что многие пользователи

(при условии, что они программисты, конечно) с готовностью смогут понять сценарий Python, а с помощью простой copy-paste смогут быстро написать свои собственные скрипты.

Для GPS существует ряд групп пользователей, которые разработали обширные модули Python, чтобы изменить поведение GPS и лучше интегрировать его в свою среду. Если бы мы выбрали использование Ada в качестве языка для таких расширений, потребовалось бы, чтобы у пользователей была полная среда разработки GPS, чтобы создавать их, повторно подключать GPS (или создавать динамически загружаемые библиотеки) и, наконец, тестировать их изменения. Для сравнения, цикл Python намного быстрее: просто отредактируйте и перезагрузите. Кроме того, часто нам удобно предоставлять быстрый и короткий сценарий Python для работы с ограничением GPS и разблокировать клиентов, пока у нас не будет времени на правильную работу на уровне Ada. Наконец, есть функции, требуемые одним клиентом, но которые не имеют смысла для других; В таком случае проще всего реализовать его с помощью короткого сценария Python по согласованию с клиентом. Если Python API поддерживается стабильным, сценарий будет продолжать работать с версии на версию GPS.

Во второй части этой серии Gem мы покажем конкретные технические сведения о том, как связать приложение с Python с помощью GNATCOLL.

Связанный со статьёй текст программы

Attached Files отсутствуют

Файлы примеров Ada Gems распространяются AdaCore и могут быть использованы или изменены для любых целей без ограничений.

Об авторе

Сведений об авторе нет.

Last Updated: 10/26/2017
Posted on: 5/9/2011

Обсуждение…