Сборки, .NET Reflection, применение плагинов в в .NET

Соломатин Д.И., solomatin@cs.vsu.ru
ПиИТ, ФКН, ВГУ

Предпосылки появления механизма **Reflection** (Отражения)

- Ранее при компиляции программ, в том числе объектно-ориентированных (C++), компилятор создавал только машинный код, который мог быть выполнен соответствующим процессором компьютера, вся информация о структуре программы (типы, функции, классы и т.п.) терялась

- Однако во многих программах возникала потребность осуществлять динамический вызов кода программы, иметь возможность «прочитать» внутреннюю структуру программы и т. п.

  - В качестве примера можно привести программу, в которую встроен интерпретатор математических выражений и необходимо обеспечить возможность использовать в таких выражениях заранее неизвестный набор функций (т.е. вызов произвольных функций, которые доступны в программе, должен осуществляться динамически по описанию функции)

Механизм Reflection позволяет

- Динамически получать любую информацию о типах и объектах, доступных в программе:
- Динамически подгружать типы данных (из внешних сборок)
- Динамически создавать экземпляры классов
- Динамически вызывать методы классов/объектов
- И т.п.

Другие типичные применения Reflection

- Сериализация данных, т.е. автоматическое сохранение объектов в какое-либо хранилище, будь то двоичный формат, xml или что-то еще
- Привязка данных, т.е., например, связывание свойств элементов управления в оконных приложениях с различными структурами данных (структурами данных бизнес-слоя сприложения)
- Автодополнение кода в современных средах программирования (информацию о доступных членах скомпилированных классов извлекается с помощью Reflection)
- Вычисление составных выражений в режиме отладки
- Автоматическое построения документации по скомпилированным классам проекта
- И т.п.

Reflection в FCL

- В .NET Framework работа с отражением доступна через классы из пространства имен System.Reflection:
  - (System.Reflection.)Type – описывает тип данных
  - MemberInfo – описывает член класса
  - MethodInfo – описывает метод класса
  - ParameterInfo – описывает параметр функции
  - PropertyInfo - описывает свойство класса
  - Assembly – описывает .NET-сборку
  - и можество других классов

Пример работы с отражением

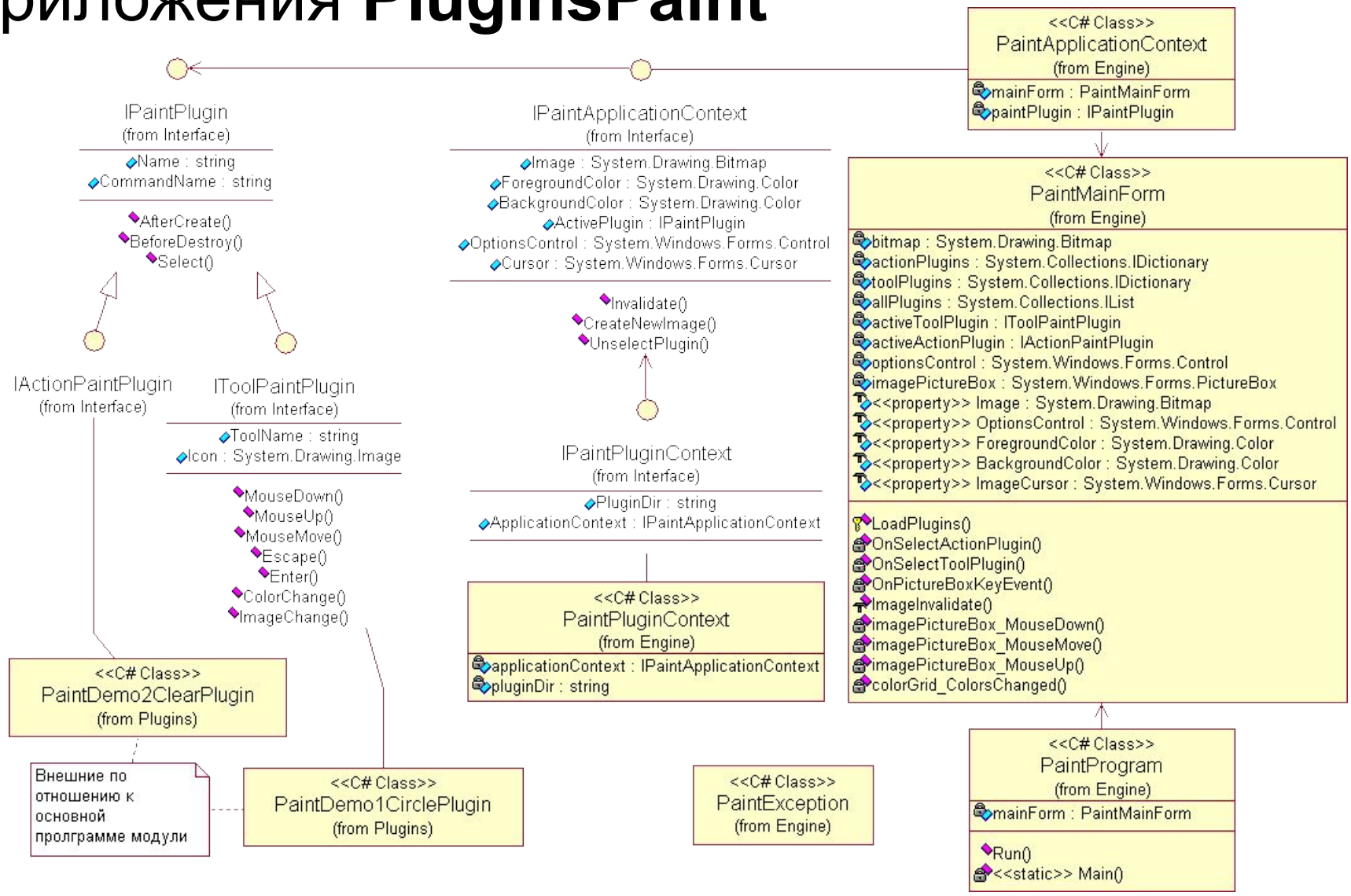
- См. пример ReflectionSamples

Понятие .NET-сборки

Понятие плагина

Приложение PluginPaint

Диаграмма классов для приложения **PluginsPaint**



Аттрибуты