Info.invoke

Вызывает метод или конструктор, представленный текущим экземпляром, используя указанные параметры.

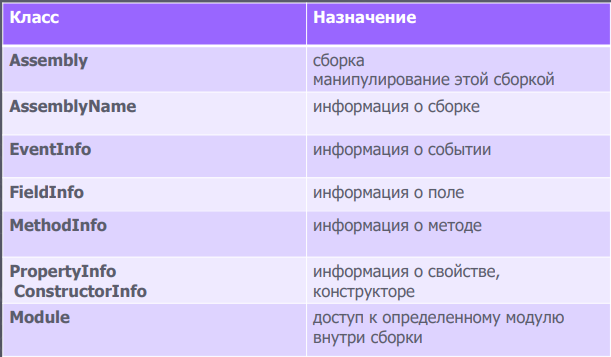
1. Что такое Рефлексия?

**Рефлексия** – процесс выявления типов во время выполнения приложения. Рефлексия позволяет: перечислять члены типа, создавать новые экземпляры объекта, запускать на выполнение члены объекта, извлекать информацию о типе, извлекать информацию о сборке, исследовать пользовательские атрибуты, примененные к типу, создавать и компилировать новые сборки.

2. Какие способы получения объекта типа Type. Приведите три способа получения типа.

1) Type t = admin.GetType ();  
2) Type t1 = Type.GetType (“System.Int32”);  
3) Type t2 = typeof (Point)

3. Охарактеризуйте классы из пр-ва имен System.Reflection



4. Как можно использовать класс System.Type? Перечислите его свойства и методы

**SystemType** – класс, позв. получ. инфу о членах типа

**Методы:**  
**FindMembers()** – возвр. массив объектов MemberInfo д-го типа  
**GetConstructors()** – конструкторы д-го типа в виде набора объектов ConstructorInfo  
**GetEvents()** – события д-го типа в виде массива объектов EventInfo  
**GetFields()** – поля д-го типа в виде массива объектов FieldInfo  
**GetInterfaces()** – реализуемые д-м типом интерфейсы в виде массива объектов Type  
**GetMembers()** – члены типа в виде массива объектов MemberInfo  
**GetMethods()** – методы типа в виде массива объектов MethodInfo  
**GetProperties()** – св-ва в виде массива объектов PropertyInfo

**Свойства:**  
**IsAbstract** – тру, если тип абстрактный  
**IsArray** – тру, если тип - массив  
**IsClass** – тру, если тип - класс  
**IsEnum** – тру, если тип - перечисление  
**IsInterface** – тру, если тип – интерфейс

5. Как динамически загрузить сборку в приложение?

при позднем связывании

6. Что такое позднее (раннее) связывание?

Позднее связывание – процесс динам. загрузки сборок и типов при работе приложения, создание экземпляров типов и работу с их эл-там

Позв. созд. расширяемые приложения, когда доп. функционал программы неизвестен и его могут разработать и подключить сторонние разработчики

**Раннее** **связывание** - это когда метод, который будет вызван, известен во время компиляции, например, вызов статического метода.

**Позднее** **связывание** – связанное с формированием кода на этапе выполнения. Если в иерархии классов встречается цепочка виртуальных методов (с помощью слов virtual, override), то компилятор строит так называемое **позднее** **связывание**

8

Форма BindingFlags метода GetMethods() чаще всего применяется для получения списка методов, определенных в классе, без дополнительного извлечения наследуемых методов.

* Задефайнить #define private public и подключить соответствующий заголовочный файл *(естественно, бессмысленно в случае PIMPL-подобных схем).*
* Напрямую обращаться по предполагаемому адресу переменной (этот способ вам уже рассказали ). Плох тем, что в зависимости от присутствия в классе vfptr и vcbl, относительное смещение может меняться, т.е способ непортабельный.
* Воспользоваться [**трюком с template.**](http://www.gotw.ca/gotw/076.htm) Самое красивое и даже в некотором плане элегантное решение.

getGrowth