# Пространства имен. Директивы препроцессора.

**№ урока:** 18 **Курс:** C# Essential

**Средства обучения:** Компьютер с установленной Visual Studio

## Обзор, цель и назначение урока

Рассмотрение пространств имен.
Создание динамически связываемых библиотек.
Рассмотрение модификаторов доступа `internal` и `protected internal`.
Рассмотрение директив препроцессора.

## Изучив материал данного занятия, учащийся сможет:

- Понимать и использовать пространства имен.
- Создавать динамически связываемые библиотеки.
- Понимать и использовать модификаторы доступа `internal` и `protected internal`.
- Понимать и использовать директивы препроцессора.

## Содержание урока

1. Пространства имен.
2. Динамически связываемые библиотеки.
3. Модификаторы доступа `internal` и `protected internal`.
4. Директивы препроцессора.

## Резюме

- **Пространства имен (namespace)** – это способ, благодаря которому .NET избегает конфликтов имен между классами.
- В программировании на C# пространства имен используются с полной нагрузкой по двум направлениям. Во-первых, платформа .NET Framework использует пространства имен для организации большинства классов. Во-вторых, объявление собственного пространства имен поможет в управлении областью действия имен классов и методов в крупных программных проектах. Для объявления пространства имен следует пользоваться ключевым словом `namespace`.
- Пространства имен имеют следующие свойства:
  1. Организация крупных проектов по созданию кода.
  2. Для их разделения используют оператор . (точка).
  3. Директива `using` исключает требование на указание имени пространства имен для каждого класса, избавляя от необходимости полной квалификации имен стереотипов.
  4. Пространство имен `global` является корневым пространством имен: `global::System` всегда будет ссылаться на пространство имен платформы .NET Framework System.
- Пространства имен и типы имеют уникальные названия, описываемые полными именами, показывающими логическую иерархию. Например, инструкция A.B подразумевает, что A – это имя пространства имен или типа, а B – это вложенный в него тип. Ключевое слово namespace используется для объявления области действия. Возможность создавать области действия в рамках проекта помогает организовывать код и позволяет создавать глобально уникальные типы.
- Директива `using` позволяет создавать псевдонимы пространства имен или типа. Это называется *директива* `using alias`.
- Ключевое слово `using` также используется для создания операторов `using`, которые обеспечивают правильную обработку объектов `IDisposable`, например, файлов и шрифтов.



CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- Директива `using alias` не может иметь открытый универсальный тип с правой части. Например, невозможно создать `using alias` для `List<T>`, но можно создать для `List<int>`.
- Технически, допускается создание нескольких пространств имен с одним именем. Логически, несколько одноименных пространств имен, объединяются в одно пространство имен. Но недопустимо иметь в одноименных пространствах имен, одноименные стереотипы.
- Во вложенных пространствах имен допустимо иметь, одноименные стереотипы.
- Два пространства имен одного уровня вложенности, не предоставляют доступа одно другому, к своим стереотипам, без импорта.
- В случае наличия одноименных стереотипов во вложенном и во внешнем пространствах имен, происходит сокрытие имени стереотипа внешнего пространства имен. Обращение к одноименному стереотипу внешнего пространства имен, потребует полной квалификации имени стереотипа.
- В случае отсутствия импорта пространства имен System, полные имена базовых типов оказываются недоступными. Доступными оказываются только псевдонимы типов.
- Ключевое слово `internal` является модификатором доступа для типов и членов типов. Внутренние типы или члены доступны только внутри файлов в одной и той же сборке.
- Доступ к типам или членам с модификатором доступа `protected internal` может осуществляться из текущей сборки или из типов, которые являются производными от содержащего их класса.
- Внутренний доступ чаще всего используется в разработке на основе компонентов, так как он позволяет группе компонентов взаимодействовать в закрытой форме, не открывая доступ остальной части кода приложения. Например, структура для построения графических пользовательских интерфейсов может предоставлять классы `Control` и `Form`, взаимодействующие при помощи членов с внутренним доступом. Так как эти члены являются закрытыми, они не предоставляются коду, использующему структуру.
- При обнаружении компилятором C# директивы `#if`, за которой далее следует директива `#endif`, компиляция кода между двумя директивами выполняется только в том случае, если определен указанный символ.
- Оператор `#if`, вместе с операторами `#else`, `#elif`, `#endif`, `#define` и `#undef`, позволяет включать или исключать код на основе существования одного или нескольких символов. Это особенно полезно при компиляции кода для построения отладки или при компиляции для определенной конфигурации.
- Директива `#region` позволяет указать блок кода, который можно разворачивать и сворачивать с помощью функции структурирования в редакторе кода Visual Studio. В больших файлах кода очень удобно сворачивать или скрывать одну или несколько областей, чтобы не отвлекать внимание от той части файла, над которой в настоящее время идет работа.

## Закрепление материала

- Что такое пространство имен?
- Что такое ключевое слово `using`?
- Что такое ключевое слово `namespace`?
- Могут ли пространства имен быть вложенными одно в другое?
- Что такое ключевое слово `alias`?
- Что такое модификатор доступа `internal`?
- Что такое модификатор доступа `protected internal`?
- Какие директивы препроцессора вы знаете, для чего они используются?

## Дополнительное задание

Задание 1
Создайте собственное пространство имен `MyNamespace` с классом `MyClass` и подключите его в другом приложении.



CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Задание 2
Выучите основные конструкции и понятия, рассмотренные на уроке.

## Самостоятельная деятельность учащегося

Задание 1
Используя пример выполненного домашнего задания 3 из 15 урока, реализуйте возможность подключения вашего пространства имен и работы с `MyDictionary`<TKey,TValue> аналогично экземпляру класса `Dictionary`<TKey,TValue>.

Задание 2
Создайте класс с методом помеченным модификатором доступа `public`. Докажите, что к данному методу можно обратиться не только из текущей сборки, но и из производного класса внешней сборки.

Задание 3
Зайдите на сайт MSDN.
Используя поисковые механизмы MSDN, найдите самостоятельно описание темы по каждому примеру, который был рассмотрен на уроке, так, как это представлено ниже, в разделе «Рекомендуемые ресурсы», описания данного урока. Сохраните ссылки и дайте им короткое описание.

## Рекомендуемые ресурсы

MSDN: Пространства имен (Руководство по программированию на C#)
http://msdn.microsoft.com/ru-ru/library/0d941h9d.aspx

MSDN: Директива `using` (Справочник по C#)
http://msdn.microsoft.com/ru-ru/library/sf0df423.aspx

MSDN: Ключевое слово `namespace` (Справочник по C#)
http://msdn.microsoft.com/ru-ru/library/z2kcy19k.aspx

MSDN: Директивы препроцессора
http://msdn.microsoft.com/ru-ru/library/ed8yd1ha.aspx



CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com