



Глава 1. Введение в С#

Глава 2. Основы программирования на С#

Глава 3. Классы. Объектно-ориентированное программирование.

Глава 4. Обработка исключений

Глава 5. Делегаты, события и лямбды

Глава 6. Интерфейсы

Глава 7. Дополнительные возможности ООП в С#

Глава 8. Объектно-ориентированное программирование. Практика

Глава 9. Коллекции

Глава 10. Работа со строками

Глава 11. Работа с датами и временем

Глава 12. Дополнительные

Работа с XML в С#

XML-документы

Последнее обновление: 14.10.2019



На сегодняшний день XML является одним из распространенных стандартов документов, который позволяет в удобной форме сохранять сложные по структуре данные. Поэтому разработчики платформы .NET включили в фреймворк широкие возможности для работы с XML.

Прежде чем перейти непосредственно к работе с XML-файлами, сначала рассмотрим, что представляет собой xml-документ и как он может хранить объекты, используемые в программе на с#.

Например, у нас есть следующий класс:

```
1 class User
2 {
3     public string Name { get; set; }
4     public int Age { get; set; }
5     public string Company { get; set; }
6 }
```

В программе на С# мы можем создать список объектов класса User:

```
1 User user1 = new User { Name = "Bill Gates", Age = 48, Company = "Microsoft" };
2 User user2 = new User { Name = "Larry Page", Age = 42, Company = "Google" };
3 List<User> users = new List<User> { user1, user2 };
```

классы и структуры
.NET

Глава 13.
Многопоточность

Глава 14.
Параллельное
программирование и
библиотека TPL

Глава 15.
Асинхронное
программирование

Глава 16. LINQ

Глава 17. Parallel LINQ

Глава 18. Рефлексия

Глава 19. Dynamic
Language Runtime

Глава 20. Сборка
мусора, управление
памятью и указатели

Глава 21. Работа с
потокami и файловой
системой

Глава 22. Работа с
JSON

Глава 23. Работа с
XML

XML-Документы

Работа с XML с
помощью
System.Xml

Изменение XML-
документа

XPath

Чтобы сохранить список в формате xml мы могли бы использовать следующий xml-файл:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <users>
3   <user name="Bill Gates">
4     <company>Microsoft</company>
5     <age>48</age>
6   </user>
7   <user name="Larry Page">
8     <company>Google</company>
9     <age>48</age>
10  </user>
11 </users>
```

XML-документ объявляет строка `<?xml version="1.0" encoding="utf-8" ?>`. Она задает версию (1.0) и кодировку (utf-8) xml. Далее идет собственно содержимое документа.

XML-документ должен иметь один единственный корневой элемент, внутри которого помещаются все остальные элементы. В данном случае таким элементом является элемент `<users>`. Внутри корневого элемента `<users>` задан набор элементов `<user>`. Вне корневого элемента мы не можем разместить элементы `user`.

Каждый элемент определяется с помощью открывающего и закрывающего тегов, например, `<user>` и `</user>`, внутри которых помещается значение или содержимое элементов. Также элемент может иметь сокращенное объявление: `<user />` - в конце элемента помещается слеш.

Элемент может иметь вложенные элементы и атрибуты. В данном случае каждый элемент `user` имеет два вложенных элемента `company` и `age` и атрибут `name`.

Атрибуты определяются в теле элемента и имеют следующую форму: `название="значение"`. Например, `<user name="Bill Gates">`, в данном случае атрибут называется `name` и имеет значение `Bill Gates`

Внутри простых элементов помещается их значение. Например, `<company>Google</company>` - элемент `company` имеет значение `Google`.

Названия элементов являются регистрозависимыми, поэтому `<company>` и `<COMPANY>` будут представлять разные элементы.

LinQ to Xml.
Создание Xml-
документа

Выборка элементов
в LINQ to XML

Изменение
документа в LINQ to
XML

Сериализация в
XML. XmlSerializer

Глава 24. Процессы и
домены приложения

Глава 25. Валидация
модели

Таким образом, весь список Users из кода C# сопоставляется с корневым элементом `<users>`, каждый объект User - с элементом `<user>`, а каждое свойство объекта User - с атрибутом или вложенным элементом элемента `<user>`

Что использовать для свойств - вложенные элементы или атрибуты? Это вопрос предпочтений - мы можем использовать как атрибуты, так и вложенные элементы. Так, в предыдущем примере вполне можно использовать вместо атрибута вложенный элемент:

```
1 <?xml version="1.0" encoding="utf-8" ?>
2 <users>
3   <user>
4     <name>Bill Gates</name>
5     <company>Microsoft</company>
6     <age>48</age>
7   </user>
8   <user>
9     <name>Larry Page</name>
10    <company>Google</company>
11    <age>48</age>
12  </user>
13 </users>
```

Теперь рассмотрим основные подходы для работы с XML, которые имеются в C#.

[Назад](#) [Содержание](#) [Вперед](#)



ТАКЖЕ НА METANIT.COM

Введение в корутины

2 месяца назад • 1 коммент...

Введение в корутины в языке программирования Kotlin, асинхронность, ...

Google представил новую ОС - Fuchsia

2 месяца назад • 2 коммент...

Google представил новую ОС - Fuchsia для устройства Google ...

Первое графическое приложение на ...

2 месяца назад • 1 коммент...

Первое графическое приложение на Rust для Windows 10, проект Rust ...

Удаление данных в MySQLi

2 месяца назад • 1 коммент...

Удаление данных в БД MySQL в языке PHP с помощью библиотеки ...

7 Комментариев

metanit.com



Политика конфиденциальности Disqus

1

Войти ▾

Рекомендовать 7

Твитнуть

Поделиться

Лучшее в начале ▾



Присоединиться к обсуждению...

ВОЙТИ С ПОМОЩЬЮ

ИЛИ ЧЕРЕЗ DISQUS (?)

Имя **Roman Kulebyakin** • 4 года назад

Как можно считать данные из Xml, если теги на русском языке?

1 ^ | ▾ 2 • Ответить • Поделиться ›

**Андрей Болконский** • 5 лет назад

чем xml лучше json?

^ | ▾ 1 • Ответить • Поделиться ›

**Metanit** Модератор ➔ Андрей Болконский • 5 лет назад

он не лучше

6 ^ | ▾ • Ответить • Поделиться ›

**Azzger** ➔ Metanit • 4 года назад

Я позволю себе дополнить предыдущий вопрос:

1. В чём преимущество xml, а в чём преимущество json?
2. В каких случаях лучше использовать, например, сериализацию, в xml, а в каких в json?

^ | ▾ • Ответить • Поделиться ›

**Dan** ➔ Azzger • 4 года назад

XML несравнимо удобнее использовать когда нужно установить какую-то структуру объектов (например, структура объектов в WPF описывается в XAML)