# برنامه نویسی پیشرفته C#

۲۸ مهر ۹۸

ملکی مجد

# Exception and error

- اتفاق های بد گاهی می افتد
  - خطاهای زیادی ممکن است در هنگام اجرای کُد رخ بدهد
  - برای مثال کاربر رفتار غیرقابل پیش بینی انجام دهد
  - برنامه شما کنار تعدادی برنامه دیگر در حال اجراست، مشکلاتی مربوط به حافظه یا شبکه رخ بدهد

- راه حل سیستم های قدیمی :
  - Special global variable

- راه حل C# و اکثر زبان های جدید شی گرا
  - Exception

# نوشتن برنامه آگاه به استثناها

- try{} catch{} finally{}

- کد مورد نظر را در بدنه مربوط به try می نویسیم ( دو حالت رخ می دهد)
  1. کد به طور کامل بدون ایجاد exception اجرا شود
  2. حالت خطا رخ بدهد، اجرای کد متوقف شده، از بدنه try خارج می شود ( یک exception تولید شده است که می تواند در قسمت catch مدیریت شود)

- مدیریت Exception
  - انواع مختلف داریم که برای هر کدام می توانیم یک catch بنویسیم.
  - بدنه catch بلافاصله بعد از try می آید.
  - متغیری که تعریف می کنیم حاوی اطلاعات مفیدی در مورد خطای ایجاد شده است.

# مثال

```
int age;
try
{
    age = int.Parse(ageString);
    Console.WriteLine("Thank you");
}
catch
{
    Console.WriteLine("Invalid age value");
}
```

# مثال – استفاده از Exception properties

```
int age;
try
{
    age = int.Parse(ageString);
    Console.WriteLine("Thank you");
}
catch (Exception e)
{
    // Get the error message out of the exception
    Console.WriteLine(e.Message);
}
```

# مثال

```
try
{
    int leftHandSide = int.Parse(lhsOperand.Text);
    int rightHandSide = int.Parse(rhsOperand.Text);
    int answer = doCalculation(leftHandSide, rightHandSide);
    result.Text = answer.ToString();
}
catch (FormatException fEx)
{
    // Handle the exception
    ...
}
```

# Unhandled exceptions

***int.Parse* statement would throw an *OverflowException***

```
try
{
    int leftHandSide = int.Parse(lhsOperand.Text);
    int rightHandSide = int.Parse(rhsOperand.Text);
    int answer = doCalculation(leftHandSide, rightHandSide);
    result.Text = answer.ToString();
}
catch (FormatException fEx)
{
    // Handle the exception
    ...
}
```

# Unhandled exceptions

- If a matching *catch* handler is eventually found, the handler runs and execution continues with the first statement that follows the *catch* handler in the catching method.

- If, after cascading back through the list of calling methods, the runtime is unable to find a matching *catch* handler, the program will terminate with an unhandled exception

```
try
{
    int leftHandSide = int.Parse(lhsOperand.Text);
    int rightHandSide = int.Parse(rhsOperand.Text);
    int answer = doCalculation(leftHandSide, rightHandSide);
    result.Text = answer.ToString();
}
catch (FormatException fEx)
{
    //...
}
catch (OverflowException oEx)
{
    //...
}
```

# Catching multiple exceptions

- *inheritance hierarchies*

- runtime uses the first handler it finds that matches the exception and the others are ignored

```
try
{
    // Exceptions at this level will be caught by the
    // "outer" catch clause

    try
    {
        // Exceptions at this level will be caught by the
        // "inner" catch clause
    }
    catch (Exception inner)
    {
        // This is the "inner" catch clause
    }

    // Exceptions at this level will be caught by the
    // "outer" catch clause
}
catch (Exception outer)
{
    // This is the "outer" catch clause
}
```

# Exception filters

```
catch (Exception ex) when (ex.GetType() != typeof(System.OutOfMemoryException))
{
    // Handle all previously uncaught exceptions except OutOfMemoryException
}
```

# finally{}

- کارهایی هست که برنامه در هر صورت باید انجام بدهد (چه exception ایجاد شود و چه نشود)
  - مانند بستن فایل، آزاد کردن منبع ها

- وقتی جریان برنامه وارد catch می شود،
  - ممکن است با return از متد خارج شویم
  - ممکن است مجددا exception جدیدی throw شود
  - ...

- بدنه finally در هر صورت بعد از try و catch اجرا می شود.

```
try
{
    // Code that might throw an exception
}
catch (Exception outer)
{
    // Code that catches the exception
}
finally
{
    // Code that is obeyed whether an exception
    // is thrown or not
}
```

# Throwing an Exception

- throw new Exception("Boom");

- می توان پیامی (به صورت رشته) برای exception تنظیم کرد.
- اگر کُد در ساختار try{}catch{} (متناسب) نباشد، برنامه با خطای زمان اجرا خاتمه پیدا می کند.

# استفاده از exception

- در مواقعی که نمی توان کار دیگری انجام داد و باید مدیریت خطا را به قسمتی دیگری بفرستید.

- برای مثال اگر کاربر ورودی نامناسب داد، می توان بدون ایجاد خطا (بدون استفاده از exception) مجددا از او درخواست وارد کردن اطلاعات داشت.