Изключения в С#

**1.Изключения:**

Докато програмираме ние описваме постъпково какво трябва да направи компютъра (поне в императивното програмиране, за което става дума в тази книга, е така) и в повечето случаи разчитаме на нормалното изпъл­нение на програмата. В по-голямата част от времето програмите следват този норма­лен ход на изпълнение, но съществуват и изключения от това правило. Да приемем, че искаме да прочетем файл и да покажем съдър­жанието му на екрана. Нека файлът се намира на отдалечен сървър и нека по време на отварянето се случи така, че връзката до този сървър пропадне и файлът се зареди само отчасти. Програмата няма да може да се изпълни нормално и да покаже съдържанието на целия файл на екрана. В този случай имаме изключение от правилното (нормалното) изпълнение на програмата и за него трябва да се сигнализира на потребителя и/или администратора.

Изключение (exception) в програмирането в общия случай представ­лява уведомление за дадено събитие, нарушаващо нормалната работа на една програма. Изключенията дават възможност необичайните събития да бъдат обработвани и програ­мата да реагира на тях по някакъв начин. Когато възникне изключение, конкрет­ното състояние на програмата се запазва и се търси обработчик на изключението (exception handler).

Изключенията се предизвикват или "хвърлят" (throw an exception**)** от програмен код, който трябва да сигнализира на изпълняващата се прог­рама за грешка или необичайна ситуация. Например ако се опитваме да отворим файл, който не съществува, кодът, който отваря файла, ще установи това и ще хвърли изключение с подходящо съобщение за грешка.

Изключенията са една от основните парадигми на обектно-ориентираното програмиране, което е описано подробно в темата "Принципи на обектно-ориентираното програмиране".

**2. Прихващане на изключения в C#**

След като един метод хвърли изключение, средата за изпълнение търси код, който евентуално да го прихване и обработи. За да разберем как действа този механизъм ще разгледаме понятието стек на извикване на методите. Това е същият този стек, в който се записват всички промен­ливи в програмата, параметрите на методите и стойностните типове.

Всяка програма на .NET започва с Main(…) метод. В него може да се извика друг метод – да го наречем "Метод 1", който от своя страна извиква "Метод 2" и т.н., докато се извика "Метод N".

Когато "Метод N" свърши работата си, управлението на програмата се връща към предходния метод и т. н., докато се стигне до Main(…) метода. След като се излезе от него, завършва и цялата програма.

Общият принцип е, че когато се извиква нов метод той се добавя най-отгоре в стека, а като завърши изпълнението му, той се изважда от стека. Така в стека за изпълнение на програмата във всеки един момент стоят всички методи, извикани един от друг – от началния метод Main() до най-последния извикан метод, който в този момент се изпълнява.

**3. Хвърляне на изключения (конструкцията throw):**

Изключения в C# се хвърлят с ключовата дума throw, като първо се създава инстан­ция на изключението и се попълва нужната информация за него. Изключенията са обикновени класове, като единственото изискване за тях е да наследяват System.Exception.

В .NET Framework има два типа от изключения: изключения генерирани от дадена програма (ApplicationException) и изключения генерирани от средата за изпълнение (SystemException). Всяко едно от тези изключения включва собствена йерархия от изключения-наследници.

Във визуалните (GUI) приложения изключенията, които не могат да бъдат обработени (или най-общо казано грешките), трябва да се показват на потреби­теля под формата на диалогов прозорец съ­държащ опи­сание, съобразено с позна­нията на потребителите:

В конзолните приложения най-често грешката се из­писва на конзолата във вид на stack trace, въпреки че това може и да не е най-удобния за крайния потребител начин за уведомление за проблеми.

При уеб приложения грешката се визуализира като червен текст в началото на страницата или около след полето, за което се отнася.

Както можете сами да си направите извода, при различните приложения изключенията и грешките се обработват по различни начини. По тази причина има много препоръки кои изключения да се хванат и кои не и как точно да се визуализират съобщенията за грешки, за да не се стряскат потребителите. Нека обясним някои от тези препоръки.

Изхвърлянето на изключения от Main() метода по принцип не е жела­телно. Вместо това се препоръчва всички изклю­чения да бъдат прихва­нати и обработени. Изхвърлянето на изключения от Main() метода все пак е възможно, както от всеки друг метод:

Всички изключения изхвърлени от Main() метода се прихващат от самата среда за изпълнение (.NET CLR) и се обработват по един и същ начин – пълният stack trace на изключението се изписва на конзолата или се визуализира по друг начин. Такова изхвър­ляне на изключенията, възник­ващи в Main() метода е много удобно, когато пишем кратка програмка набързо и не искаме да обработваме евентуално възникващите изключе­ния. Това е бягане от отговорност, което се прави при малки прости програмки, но не трябва да се случва при големи и сериозни приложения.