**27.Правила за работа с изключения в .NET среда**

Правила за работа с изключения

1. разработвате библиотека: ако прихванете всички изкл. , как

разработващия приложение с библиотеката ще знае че нещо се е случило

2. разработвате библиотека с типове – не винаги знаете кое е грешка, кое не. Оставете това на викащия

3. Избягвайте код , прихващащ всичко: catch(System.Exception) {………}

4. Ако операция е частично завършена изключение и следва възстановяване в начално съст.: най-добре прихванете всичко, възстановете и уведомете (с друго изкл. ) викащата страна.

5. След прихващане и обработка на изключение, често е добре да уведомите извикващия: подавате същото (само с throw) или друго изключение (това е начина за преобразуваме изключението от нещо специфично, към общоразбираемо за потребител).

Необработвани съобщения (такива, които никой catch не разпознава)

Най напред следва да се разработи единна политика за тях – напр. въведен текст се съхранява и се визуализира диалогов прозорец с информация и т.н.;

-1.При отдалечено викана процедура или web услуга или сървърно-базиран код, който подава exception, то той се изпълнява в сървърно обкръжение на try/catch. Тъи като exception обекта е сериализиран, той може да се предава през граница на Domain – т.е. обратно към клиентското приложение.

-2. В общия случай, необработени съобщения могат да се насочват за обработка към дефинирана в в приложението делегатна функция, регистрирана като

event handle от тип System.UnhandledExceptionEventHandler към стандартния тип за изключния: System.AppDomain.UnhandledException . Пример:

AppDomain.CurentDomain.UnhandledException +=

new UnhandledExceptionEventHandler(MyUnhandledExceptionFunction);

3. Необработваните изключения в приложения, базирани на Windows Forms се прихващат така: цялата WinProc ф-ия всъщност, се вика в обхващащ я автоматично try/catch.

При наличие на необработено по-долу изключение, catch блокът извиква виртуалния метод OnThreadException() дефиниран в System.Windows.Forms.Control и предефиниран в Application

Той визуализира стандартен прозорец за ‘unhandled exception’

Можете да предефинирате поведението чрез ваш метод от делегатен тип

System.Threading.ThreadExceptionEventHandler

и след това да свържете този метод с ThreadException събитието на класа Application

-4. Необработени съобщения в ASP.NET

ASP обхваща кода на приложението в собствен try блок и предопределя начин за обработка. Може да се намесите като регистрирате свой callback метод към събитие Error на класа System.Web.UI.Page или на клас System.Web.UI.UserControl

(методът може и да се вика за всяко необработено изключение от която и да е страница на

приложението – ако callback метода е свързан с Error събитие на клас

System.Web.HTTPApplication)

5. Необработени изключения в среда ASP.NET XML

Отново обхващащ кода try блок на ASP.NET подава SoapException обект. Той се сериализира в

XML вид и може да се предава към друг компютър или приложение, работещо като клиент на XML Web услугата.