به نام خدا

آموزش پنجاه و چهارم

اهداف آموزشی این قسمت عبارتند از:

۱. معرفی Exception در زبان برنامه نویسی جاوا
۲. آشنایی با دستورات try, catch, finally

در این آموزش با مفهومی تحت عنوان Exception در زبان برنامه نویسی جاوا آشنا خواهیم شد. به طور کلی در زبان جاوا به منظور مواجهه با Error ها از چیزی تحت عنوان Exception استفاده می کنیم. معادل فارسی واژه Exception برابر است با "استثناء" و در زبان جاوا به "رویدادی گفته می شود که در پروسه اجرای یک برنامه یا اپلیکیشن بوجود می آید و از اجرای طبیعی برنامه جلوگیری به عمل می آورد".

فرض کنیم متدی داریم که این وظیفه را دارا است تا کاری انجام دهد. حال در حین اجرای دستورات داخل این متد یک Error روی می دهد. در شرایطی این چنین، کاری که این متد انجام می دهد این است که شیئی تحت عنوان Exception Object می سازد که حاوی اطلاعاتی پیرامون نوع Error و همچنین زمانیکه این Error در برنامه یا اپلیکیشن رخ داده است می باشد و سپس این شیئ را تحویل سیستم می دهد. از این مرحله به بعد سیستم سعی می کند تا راه کاری برای رفع این Error بیابد. اگر سیستم بتواند دستور یا به طور کلی کدی را بیابد که بتواند این Error را رفع کند، آن دستور یا کد که اصطلاحاً Exception Handler نام دارد به رفع مشکل برنامه ما خواهد پرداخت و در غیر این صورت برنامه Crash خواهد کرد.

به طور کلی در زبان برنامه نویسی جاوا وقتی این احتمال وجود داشته باشد که ممکن است با یک Exception مواجه شویم، بایستی کد خود را داخل دستوری تحت عنوان try بنویسیم که در این صورت اگر Error هم داخل برنامه یا اپلیکیشن ما وجود داشته باشد برنامه به هیچ وجه Crash نخواهد کرد.

برای روش شدن این مطلب پروژه تحت عنوان 54th Session در محیط اکلیپس ایجاد کرده و کلاسی تحت عنوان ExceptionsInJava ایجاد می کنیم (به خاطر داشته باشیم که در حین ساخت این کلاس گزینه public static void main را تیک بزنیم):

```
public class ExceptionsInJava {

	public static void main(String[] args) {

	}

}
```

همانطور که ملاحظه می شود پس از حذف کامنت ها کد ما به صورت بالا خواهد بود. برای درک بهتر روابط مابین بخش های مختلف Exception ها کد فوق را به صورت زیر تکمیل می کنیم:

```
public class ExceptionsInJava {

	public static void main(String[] args) {

		try{
			// کدی که می خواهیم اجرا شود
		}catch(){
			//کدی که در صورت بروز مشکل می خواهیم اجرا شود
		}finally{
			//کدی که در پایان اجرای مراحل فوق می خواهیم اجرا شود
		}

	}

}
```

همانطور که در کد فوق می بینیم مابین دو {} مرتبط با دستور try بخشی از برنامه خود را می نویسیم که می خواهیم اجرا شود. حال این بخش از کد ما ممکن است که برنامه را با Error یی مواجه سازد. از این رو داخل دو {} مرتبط با دستور catch کدی را می نویسیم که در صورت بروز هر گونه مشکلی اجرا شود. در نهایت داخل دو {} مرتبط با دستور finally کدی را می نویسیم که در انتهای برنامه قصد داریم اجرا شود. حال اگر کدی که داخل دستور try است هیچ گونه مشکلی ایجاد نکرد، برنامه ما مستقیم به سراغ کدی خواهد دارد که داخل دستور finally قرار دارد و اگر هم کدی که داخل دستور try بود مشکلی ایجاد کرد، برنامه ابتدا دستور داخل catch را اجرا خواهد نمود سپس به سراغ اجرای دستور داخل finally خواهد رفت.

اگر توجه کرده باشیم خواهیم دید که مقابل دستور catch دو پرانتز قرار دارد. کاری که این دو پرانتز انجام می دهند این است که می بایست داخل آنها نوع Exception یی که قصد داریم سیستم تشخیص دهد را بنویسیم. به عبارت دیگر چیزی که می بایست داخل پرانتزها نوشت نام یک کلاس از پیش تعریف شده در API زبان جاوا است که مرتبط با Exception ها می باشد به علاوه نام شیئی که برای آن کلاس در نظر می گیریم (در ادامه آموزش به خوبی به توضیح این مسئله خواهیم پرداخت).

اکنون برای آنکه به طور عملی با کارکرد Exception ها آشنا شویم کد فوق را به صورت زیر تکمیل می کنیم:

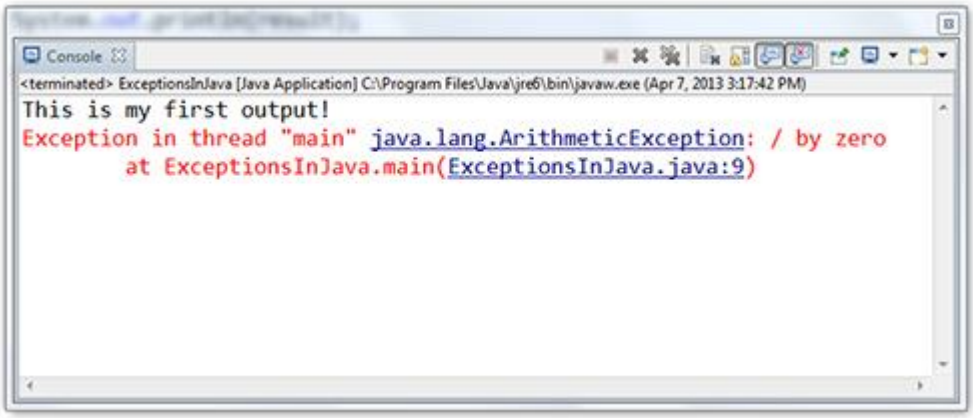
```
public class ExceptionsInJava {

	public static void main(String[] args) {

		System.out.println("This is my first output!");

		int numberOne = 10;
		int numberTwo = 0;
		int result = numberOne / numberTwo;

		System.out.println(result);

		System.out.println("This is my final output!");

	}
}
```

همانطور که در کد بالا مشاهده می کنیم در ابتدا با استفاده از دستور System.out.println عبارت This is my first output! به معنی "این اولین خروجی من است!" را در پنجره Console به نمایش در خواهیم آورد. سپس اقدام به تعریف چند متغیر از جنس int نموده که این وظیفه را دارا هستند که اعدادی از جنس عدد صحیح را در خود ذخیره سازند. متغیر اول ما numberOne به معنی "عدد شماره یک" است که مقدار اولیه آن معادل با ۱۰ است. متغیر دوم ما numberTwo به معنی "عدد شماره دو" است که مقدار اولیه آن معادل با صفر است. متغیر سوم ما result به معنی "نتیجه" نام دارد که عددی به عنوان Value آن در نظر نگرفته ایم بلکه

Value آن را حاصل تقسیم مقدار متغیر numberOne بر numberTwo قرار داده ایم (به منظور آشنایی بیشتر با اعمال ریاضیاتی در جاوا به آموزش هشتم مراجعه نمایید).

همانطور که منطق ریاضی حکم می کند ما نمی توانیم عددی همچون ده را بر صفر تقسیم کنیم. علیرغم اینکه از این نکته اطلاع داریم می خواهیم ببینم که عکس العمل ماشین مجازی جاوا یا همان JVM چیست. حال یک بار دیگر دستور System.out.println را نوشته و این بار با قرار دادن نام متغیر result داخل پرانتز این دستور، از سیستم می خواهیم که پس از به نمایش در آوردن عبارت This is my first output! اقدام به نمایش مقدار متغیر result نماید. در نهایت یک بار دیگر دستور System.out.println را نوشته و این بار عبارت This is my final output! به معنی "این خروجی پایانی من است!" را می خواهیم در پنجره Console به نمایش در آوریم.

پس از اجرای برنامه خروجی زیر مشاهده خواهد شد:



همانطور که در اجرای بالا می بینیم، اولین دستوری که نوشته بودیم بدون هیچ مشکلی اجرا شده و عبارت This is my first output! در پنجره Console به نمایش در آمده است. سیستم پس از اجرای اولین دستور به سراغ دستور دوم خواهد رفت و از آنجا که در دستور دوم از برنامه خود خواسته ایم که عدد ده را به عدد صفر تقسیم کند و این چنین عملی خارج از منطق ریاضیاتی است، از این رو برنامه ما اصطلاحاً Crash کرده و در پنجره Console همانطور که در تصویر فوق مشخص است یک Exception از نوع کلاس `ArithmeticException` در برنامه رخ داده است. واژه `Arithmetic` به معنی "محاسباتی، ریاضیاتی و ..." است و همانطور که از نام این کلاس پیدا است نوع `Exception` بوجود آمده در ارتباط با اعمال ریاضیاتی است.

حال از آنجا که برنامه ما در حین اجرای دومین دستور خود Crash کرد بنابراین از اجرای دستور سوم که همان به نمایش در آوردن عبارت This is my final output! است نیز ناتوان خواهد بود.

در آموزش آتی خواهیم دید که به چه شکل با استفاده از دستورات try, catch, finally پروژه ای که در این آموزش نوشتیم و Crash کرد را باز نویسی کرده و از Crash کردن آن جلوگیری خواهیم کرد.

پس از مطالعه این آموزش انتظار می رود بتوانیم به سؤالات زیر پاسخ بدهیم:

۱. منظور از Exception در زبان جاوا چیست؟
۲. منظور از Handle کردن یک Exception چیست؟
۳. Exception یی که مرتبط با اشکالات ریاضیاتی است چه نام دارد؟