Исключения

Основы

Для облегчения работы с ошибками большинство современных языков имеют механизм исключений. Состояние программы, когда она не может продолжать выполнение данного фрагмента кода, называется **исключительным состоянием**. Необходимо передать управление другому фрагменту, который может выполнить необходимые в данном случае действия. При возникновении исключительного состояния создается объект специального класса, который также называют **исключением**.

Все подобные объекты связаны между собой наследованием, у них есть общий предок **Throwable**, наследниками которого являются два класса: **Exception** и **Error**. Все остальные классы являются наследниками этих двух. У исключительных ситуаций, описываемых классом *Error* и его наследниками, есть общая особенность: они вызывают ошибки настолько серьезные, что пользователь ничего не может с ними поделать. Например, ошибка *OutOfMemoryError* возникает в случае, если закончилась память. При возникновении этой и других подобных ошибок продолжение работы программы невозможно. Если же возникает исключение, основанное на классе *Exception*, то пользователь имеет возможность прописать действия, необходимые для устранения последствий, и программа сможет продолжить работу.

Среди исключений, основанных на *Exception*, также есть особая группа: класс *RuntimeException* и его наследники. Такие исключения могут создаваться как программистом явно, так и виртуальной машиной самостоятельно. Типичный пример: **NullPointerException**. Данное исключение создаётся самой виртуальной машиной, если пользователь попытался вызвать метод или свойство переменной, в которой хранится *null*.

Все исключения разделяют на **checked**, то есть обрабатываемые, и **unchecked**, или необрабатываемые. Обрабатываемые исключения в языке программирования Java — это исключения, которые программисту необходимо обрабатывать явно. К ним относятся классы *Throwable*, *Exception* и все их потомки за исключением классов *RuntimeException* и *Error* вместе с наследниками этих классов.

Ключевое слово throw

Для создания исключения используется ключевое слово **throw**. За ним должен следовать объект исключения. Обычно его не описывают в начале кода программы, а создают тут же на месте с помощью new:

```
throw new NullPointerException();
```

Здесь *NullPointerException* — это класс исключения, объект которого создается. Как видно из этого примера, основой для исключений является объект исключения. Классы исключений являются дочерними от класса *Throwable*. В простейших случаях допустимо не создавать свой новый класс, а использовать непосредственно этот класс *Throwable*, хотя подобная практика не рекомендуется. Правилом хорошего тона будет создать новый класс, унаследовав его от Exception.

```
class CustomException extends Throwable {

}
```

Обработка исключений

Следует помнить, что вызов исключения используются не только для того, чтобы его создавать, но и обрабатывать. Для этого используется конструкция следующего вида:

```
try {
        операторы1;
} catch (Тип имя) {
        операторы2;
} finally {
        операторы3;
}
```

В блоке *try* содержатся операторы1, которые могут вызвать ошибку и создать исключения. В блоке «*catch*» содержатся операторы, которые вызываются при возникновении исключения. Блок «*finaly*» содержит операторы, которые должны выполняться в любых условиях, то есть независимо от того, было исключение или его не было.

Если в блоке *try* может возникнуть несколько видов исключений, для их обработки можно создать несколько идущих подряд блоков *catch*. Будет вызван тот из них, который соответствует создаваемому исключению. Также можно обойтись одним блоком *catch*, но в этом случае в нем следует создавать переменную родительского класса исключений. В этом случае «ловятся» не только исключения именно этого класса, но и исключения его потомков. Таким образом, если в *catch* в качестве типа указано *Exception*, будут «ловиться» все исключения.

Следует помнить, что блок *try* может не содержать создание исключения, а только вызывать метод, в котором это исключение возникает.

Если есть несколько видов исключений, но способ их обработки одинаковый, то можно писать так:

```
try {
операторы1;
} catch(Тип1|Тип2|Тип3 имя) {
операторы2;
}
```

Ключевое слово throws

Рассмотрим случай, когда в методе возникает исключение, и непосредственно в нем оно не обрабатывается. Иначе говоря, конструкция, приведенная выше, отсутствует. Тогда в заголовке следует указать, что этот метод может вызывать исключение, и что его

обработкой должен заниматься вызывающий метод. Для этого при объявлении метода указывается, что он «выбрасывает» (**throws**) исключение, и имя класса исключения:

```
public static void main(String[] args) throws IOException
```