متد

متدها به شما اجازه می‌دهند که یک رفتار یا وظیفه را تعریف کنید و مجموعه‌ای از کدها هستند که در هر جای برنامه می‌توان از آنها استفاده کرد. متدها دارای آرگومانهایی هستند که وظیفه متد را مشخص می‌کنند. متد در داخل کلاس تعریف می‌شود. نمی‌توان یک متد را در داخل متد دیگر تعریف کرد. وقتی که شما در برنامه یک متد را صدا می‌زنید برنامه به قسمت تعریف متد رفته و کدهای آن را اجرا می‌کند. در جاوا متدی وجود دارد که نقطه آغاز هر برنامه است و بدون آن برنامه‌ها نمی‌دانند باید از کجا شروع شوند، این متد main() نام دارد.

> پارامترها همان چیزهایی هستند که متد منتظر دریافت آنها است.
>
> آرگومان‌ها مقادیری هستند که به پارامترها ارسال می‌شوند.

گاهی اوقات دو کلمه پارامتر و آرگومان به یک منظور به کار می‌روند. ساده‌ترین ساختار یک متد به صورت زیر است :

```
returnType MethodName()
{
   code to execute;
}
```

به برنامه ساده زیر توجه کنید. در این برنامه از یک متد برای چاپ یک پیغام در صفحه نمایش استفاده شده است :

```
1  package myfirstprogram;
2
3  public class MyFirstProgram
4  {
5      static void PrintMessage()
6      {
7          System.out.println("Hello World!");
8      }
9
10     public static void main(String[] args)
11     {
12         PrintMessage();
13     }
14 }
```

```
Hello World!
```

در خطوط 5-8 یک متد تعریف کرده‌ایم. مکان تعریف آن در داخل کلاس مهم نیست. به عنوان مثال می‌توانید آن را زیر متد main() تعریف کنید. می‌توان این متد را در داخل متد دیگر صدا زد (فراخوانی کرد). متد دیگر ما در اینجا متد main() است که می‌توانیم در داخل آن نام متدی که برای چاپ یک پیغام تعریف کرده‌ایم (یعنی متد PrintMessage()) را صدا بزنیم. متد main() به صورت static تعریف شده است. برای اینکه بتوان از متد PrintMessage() در داخل متد main() استفاده کنیم باید آن را به صورت static تعریف کنیم.

کلمه static به طور ساده به این معناست که می‌توان از متد استفاده کرد بدون اینکه از کلاس نمونه‌ای ساخته شود. متد main() همواره باید به صورت static تعریف شود چون برنامه فوراً و بدون نمونه سازی از کلاس از آن استفاده می‌کند. وقتی به مبحث برنامه نویسی شیء گرا رسیدید به طور دقیق کلمه static مورد بحث قرار می‌گیرد. برنامه MyFirstProgram (مثال بالا) زمانی اجرا می‌شود که برنامه دو متدی را که تعریف کرده‌ایم را اجرا کند و متد main() به صورت static تعریف شود. درباره این کلمه کلیدی در درسهای آینده مطالب بیشتری می‌آموزیم.

در تعریف متد بالا بعد از کلمه static کلمه کلیدی void آمده است که نشان دهنده آن است که متد مقدار برگشتی ندارد. در درس آینده در مورد مقدار برگشتی از یک متد و استفاده از آن برای اهداف مختلف توضیح داده خواهد شد. نام متد ما PrintMessage() است.

به این نکته توجه کنید که در نامگذاری متد از روش پاسکال (حرف اول هر کلمه بزرگ نوشته می‌شود) استفاده کرده‌ایم. این روش نامگذاری قراردادی است و می‌توان از این روش استفاده نکرد، اما پیشنهاد می‌شود که از این روش برای تشخیص متدها استفاده کنید. بهتر است در نامگذاری متدها از کلماتی استفاده شود که کار آن متد را مشخص می‌کند مثلاً نام‌هایی مانند GoToBed یا OpenDoor.

همچنین به عنوان مثال اگر مقدار برگشتی متد یک مقدار بولی باشد می‌توانید اسم متد خود را به صورت یک کلمه سوالی انتخاب کنید مانند IsLeapyear یا IsTeenager ... ولی از گذاشتن علامت سؤال در آخر اسم متد خودداری کنید. دو پرانتزی که بعد از نام می‌آید، نشان دهنده آن است که، نام متعلق به یک متد است. در این مثال در داخل پرانتزها هیچ چیزی نوشته نشده چون پارامتری ندارد. در درسهای آینده در مورد متدها بیشتر توضیح می‌دهیم.

بعد از پرانتزها دو آکولاد قرار می‌دهیم که بدنه متد را تشکیل می‌دهد و کدهایی را که می‌خواهیم اجرا شوند را در داخل این آکولادها می‌نویسیم. در داخل متد main() متدی را که در خط 12 ایجاد کرده‌ایم را صدا می‌زنیم. برای صدا زدن یک متد کافیست نام آن را نوشته و بعد از نام پرانتزها را قرار دهیم.

اگر متد دارای پارامتر باشد باید شما آرگومانها را به ترتیب در داخل پرانتزها قرار دهید. در این مورد نیز در درسهای آینده توضیح بیشتری می‌دهیم. با صدا زدن یک متد کدهای داخل بدنه آن اجرا می‌شوند. برای اجرای متد PrintMessage() برنامه از متد main() به محل تعریف متد PrintMessage() می‌رود. مثلاً وقتی ما متد PrintMessage() را در خط 12 صدا می‌زنیم برنامه از خط 12 به خط 7، یعنی جایی که متد تعریف شده می‌رود. اکنون ما یک متد در برنامه MyFirstProgram داریم و همه متدهای این برنامه می‌توانند آن را صدا بزنند.