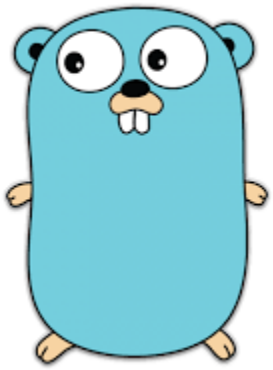


Go

به زبان ساده

تقدیم به

همه جویندگان علم

راه‌های ارتباط با نویسنده

وب سایت: www.w3-farsi.com

لینک تلگرام: https://telegram.me/ebrahimi_younes

ID تلگرام: @ebrahimi_younes

پست الکترونیکی: younes.ebrahimi.1391@gmail.com

GO چیست

زبان برنامه نویسی Go که به گولنگ یا Golang معروف می‌باشد، یک زبان برنامه نویسی است که در سال ۲۰۰۹ توسط Ken Thompson و Rob Pike و Robert Griesemer در شرکت گوگل ابداع و به صورت متن باز منتشر شد.

علاوه بر گوگل، شرکت‌های بزرگی مانند YouTube، BBC، SoundCloud و غیره از این زبان برای طراحی سیستم‌های Back-end استفاده می‌کنند.

می‌توان گفت که Go با ظرافت فراوان قدرت و سرعت زبانی مثل C را با سهولت و سادگی زبانی مثل Python ترکیب کرده. به همین دلیل قادر است طیف بسیار وسیعی از برنامه‌ها را پوشش دهد، از برنامه‌های سیستمی گرفته تا برنامه‌های ساده چند خطی.

Go زبانی است از خانواده C و به همین دلیل برنامه نویسانی که با C++، C، Java، C#، PHP، JavaScript و ... آشنایی دارند، بسیار راحت این زبان را یاد خواهند گرفت.

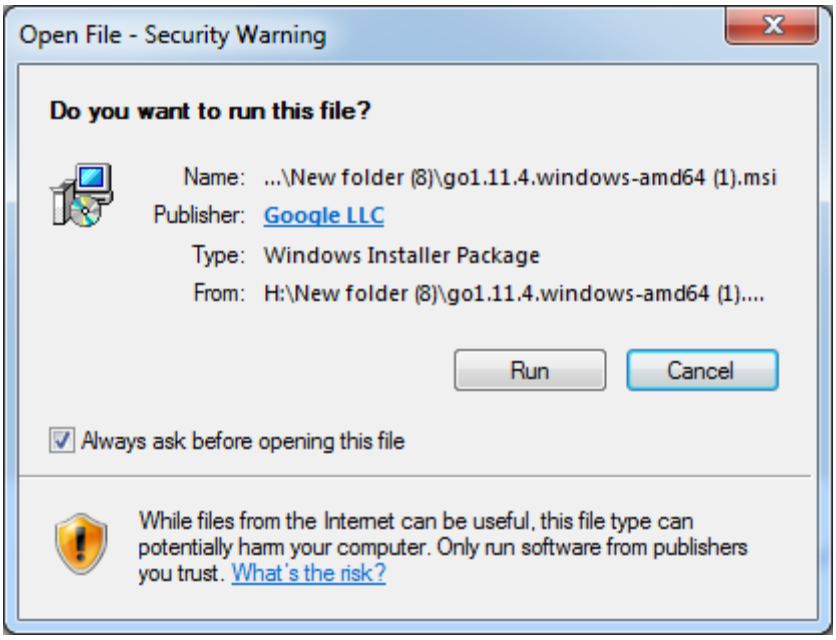
Go یک زبان برنامه‌نویسی همه منظوره با امکانات پیشرفته و دستور زبان شفاف می‌باشد. بخاطر پشتیبانی از گستره بسیاری از پلتفرم‌ها، کتابخانه‌های قدرتمند مستند سازی شده و تمرکز روی اصول مهندسی نرم‌افزار، Go یکی از ایده‌آل ترین زبان‌ها برای یادگیری به عنوان اولین زبان برنامه‌نویسی می‌باشد.

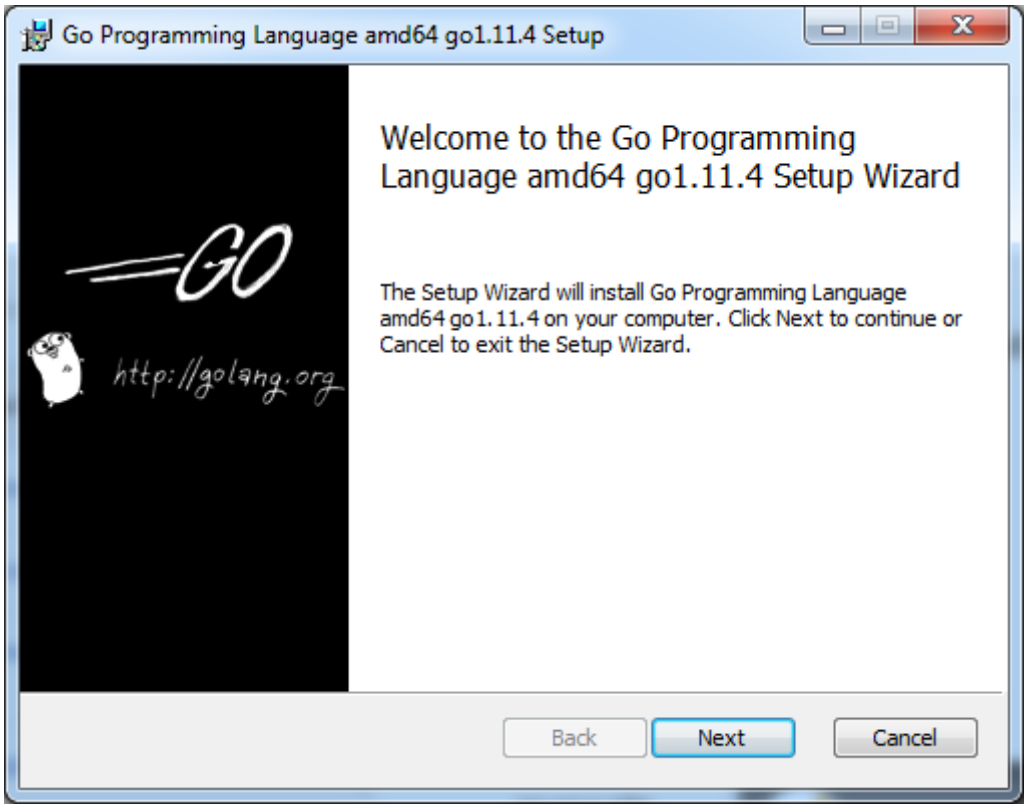
نصب و راه اندازی Go

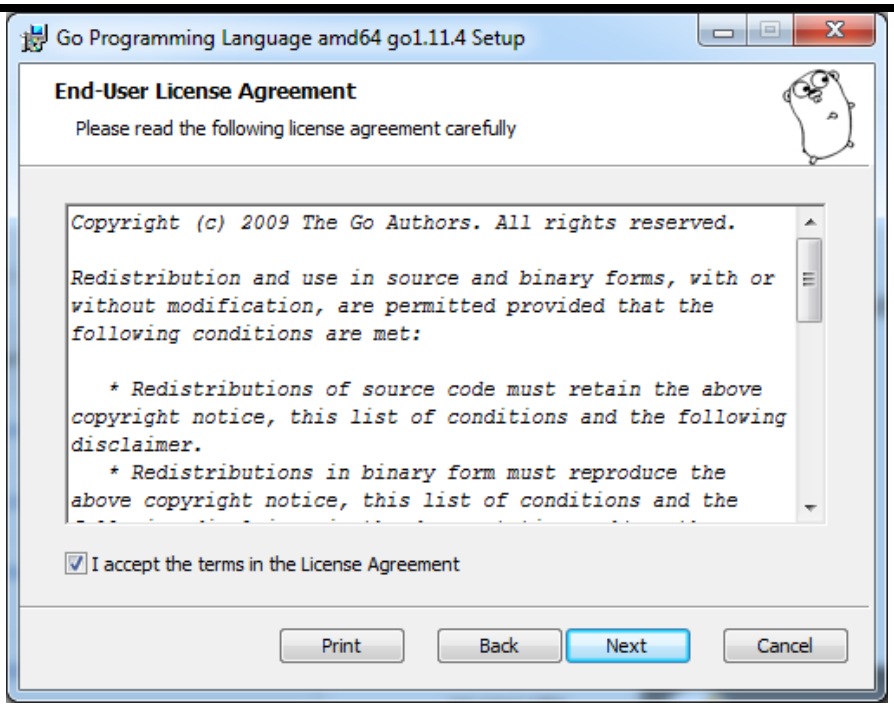
برای کدنویسی به زبان Go و همچنین اجرای کدهای این زبان به دو ابزار احتیاج دارید. یکی از آنها، یک ویرایشگر متن ساده مانند NotePad خود ویندوز و دیگری کامپایلر زبان Go می‌باشد. این کامپایلر را می‌توانید از لینک زیر دانلود کنید:

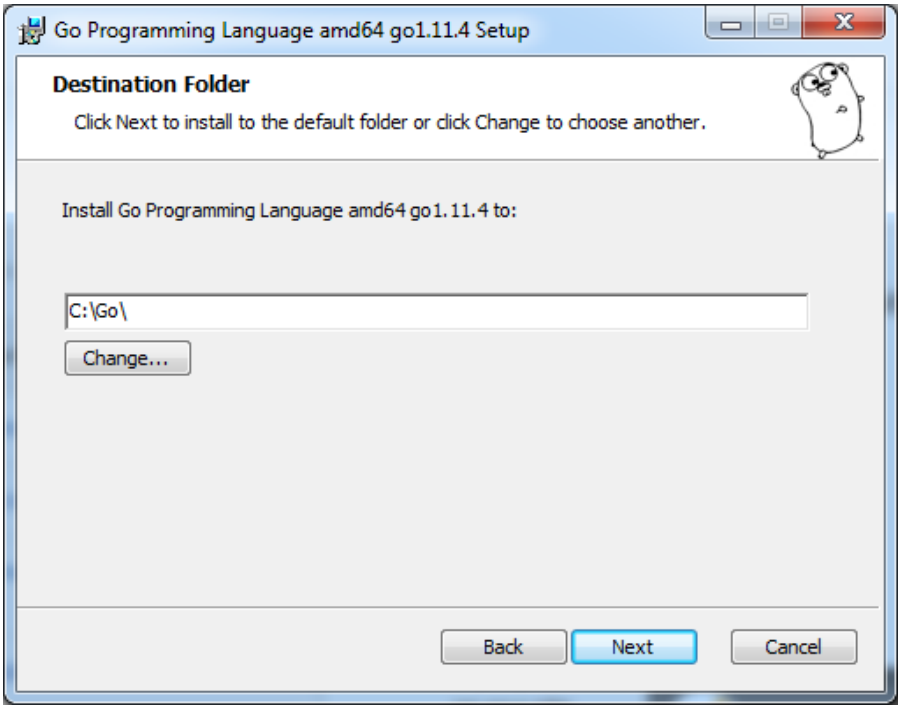
```
http://dl.w3-farsi.com/Software/GO/go1.11.4.windows-amd64.msi
```

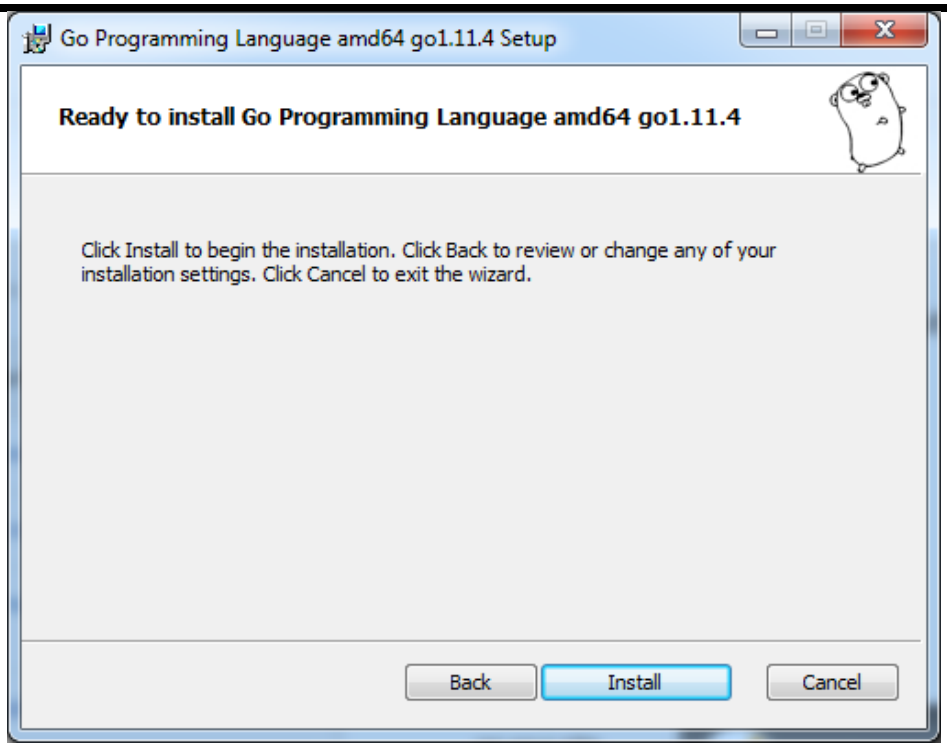
بعد از دانلود فایل بالا، با دوبار کلیک بر روی آن و زدن چند دکمه Next یا Yes، که مراحل آن در زیر نمایش داده شده است، کامپایلر Go نصب می‌شود:

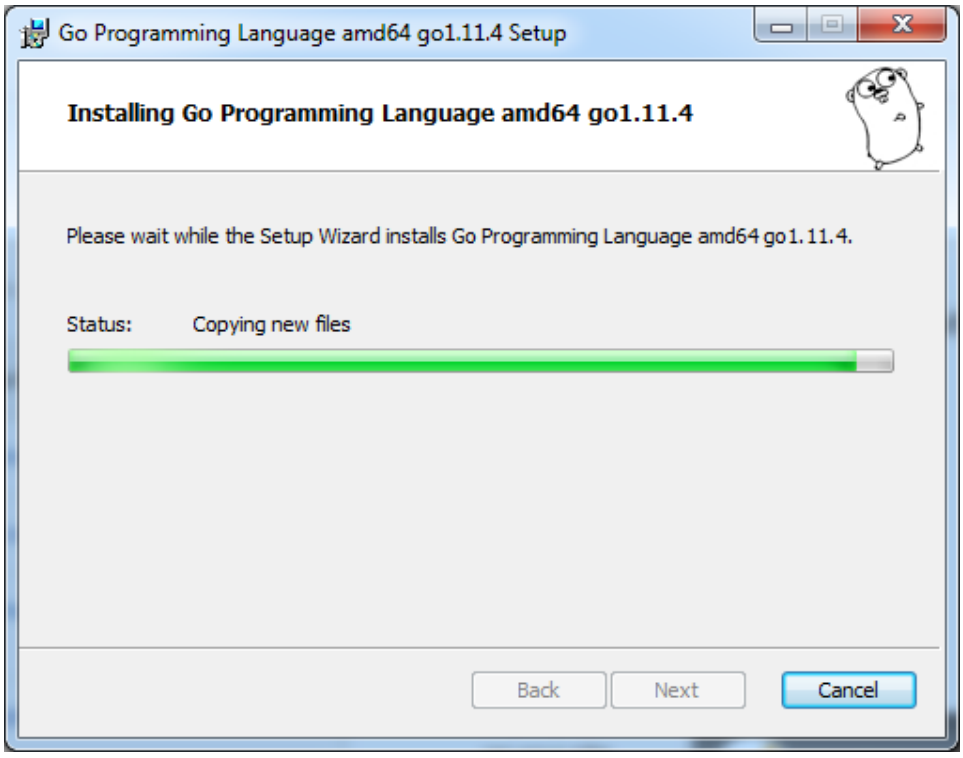

Open File - Security Warning
Do you want to run this file?
Name: ...\New folder (8)\go1.11.4.windows-amd64 (1).msi
Publisher: Google LLC
Type: Windows Installer Package
From: H:\New folder (8)\go1.11.4.windows-amd64 (1)....
Run
Cancel
Always ask before opening this file
While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

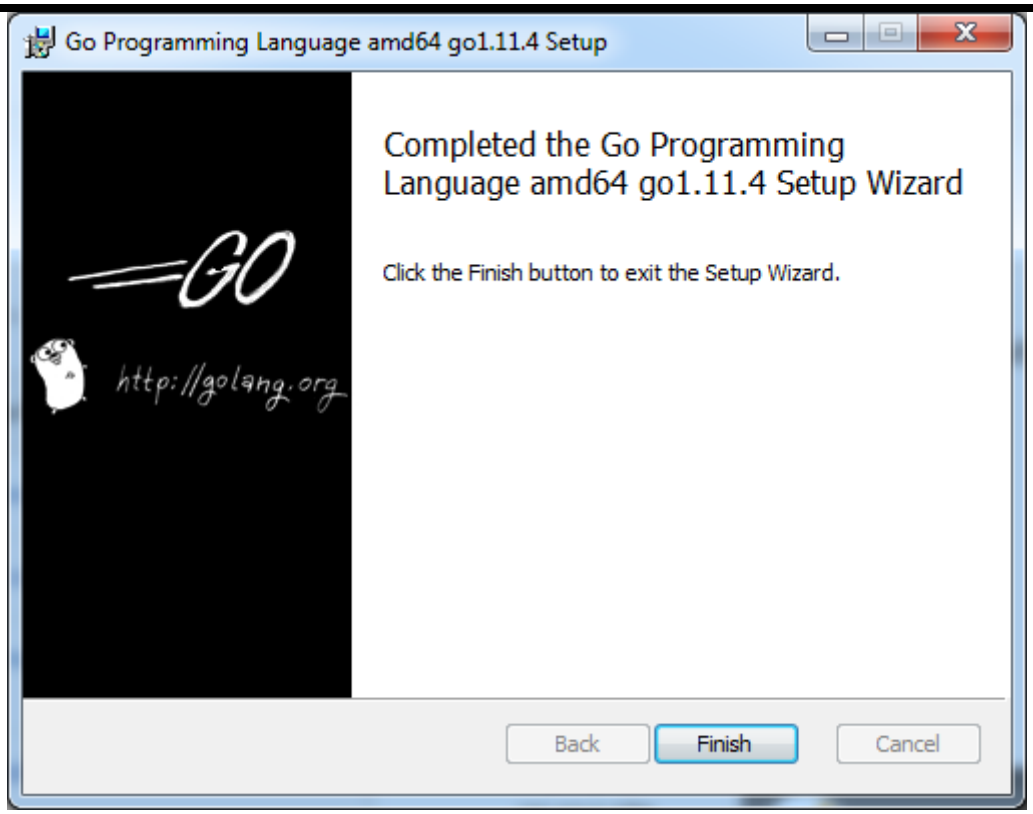

Go Programming Language amd64 go1.11.4 Setup
Welcome to the Go Programming Language amd64 go1.11.4 Setup Wizard
The Setup Wizard will install Go Programming Language amd64 go1.11.4 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.
Back
Next
Cancel


Go Programming Language amd64 go1.11.4 Setup
End-User License Agreement
Please read the following license agreement carefully
Copyright (c) 2009 The Go Authors. All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the
I accept the terms in the License Agreement
Print
Back
Next
Cancel


Go Programming Language amd64 go1.11.4 Setup
Destination Folder
Click Next to install to the default folder or click Change to choose another.
Install Go Programming Language amd64 go1.11.4 to:
C:\Go\
Change...
Back
Next
Cancel


Go Programming Language amd64 go1.11.4 Setup
Ready to install Go Programming Language amd64 go1.11.4
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel


Go Programming Language amd64 go1.11.4 Setup
Installing Go Programming Language amd64 go1.11.4
Please wait while the Setup Wizard installs Go Programming Language amd64 go1.11.4.
Status: Copying new files
Back
Next
Cancel



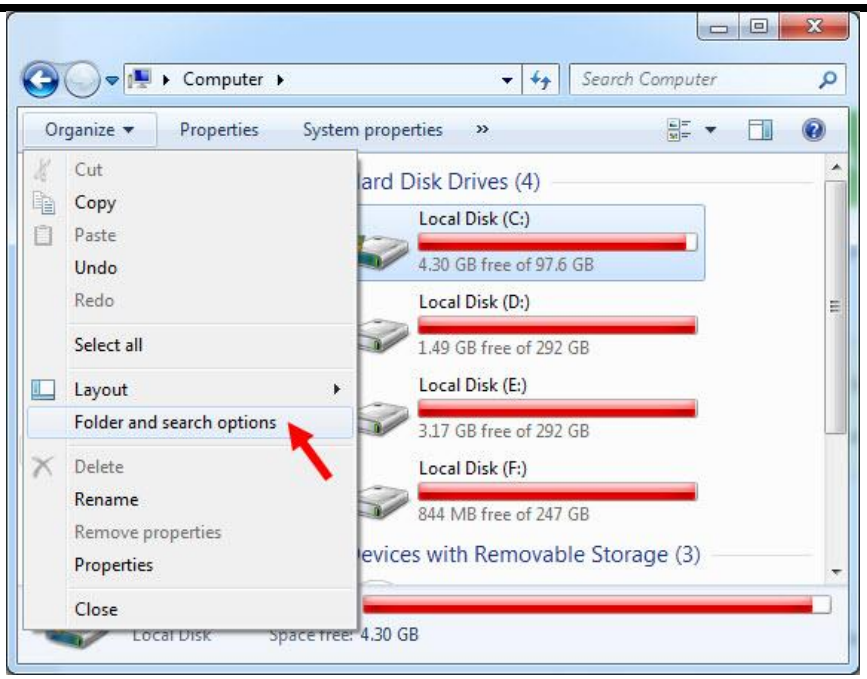
بعد از طی مراحل بالا، شما به راحتی می‌توانید، کدنویسی خود را شروع کنید. در درس بعد شما را با نحوه اجرای کدهای زبان Go آشنا می‌کنم.

ساخت یک برنامه ساده در Go

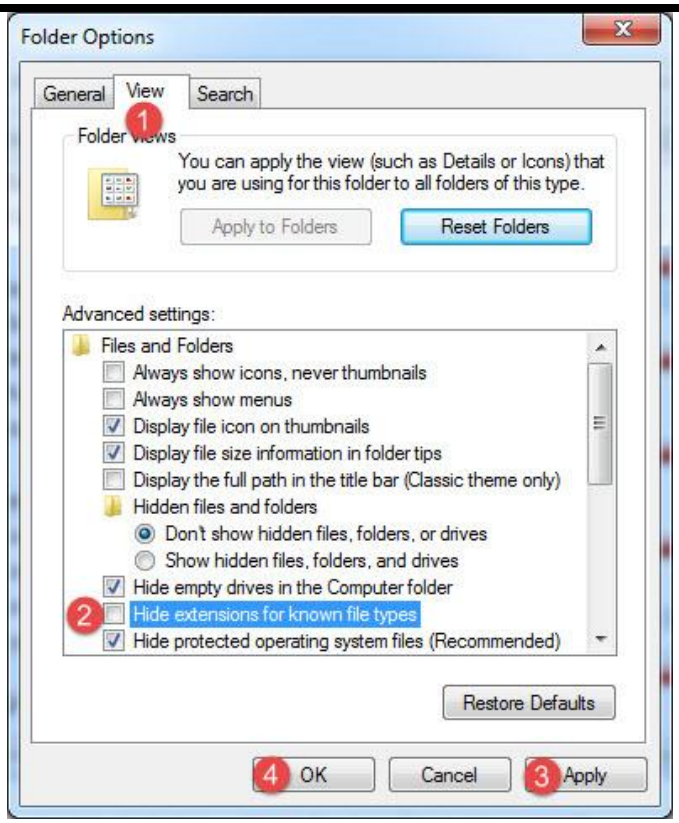
اجازه بدهید یک برنامه بسیار ساده به زبان Go بنویسیم. این برنامه یک پیغام را نمایش می‌دهد. در این درس می‌خواهم ساختار و دستور زبان یک برنامه ساده GO را توضیح دهم. قبل از ایجاد برنامه به این نکته خیلی مهم توجه کنید:

> در نوشتن این برنامه و برنامه‌های آتی، به حروف بزرگ و کوچک، توجه کنید. چون Go به بزرگ و کوچک بودن حروف حساس است.

یکی از تنظیماتی که قبل از شروع این درس توصیه می‌کنیم که اعمال کنید این است که پسوند فایل‌ها داخل ویندوز را قابل مشاهده کنید. برای این کار به `My Computer` رفته و به صورت زیر از منوی `Organize` گزینه `Folder and search options` را بزنید:

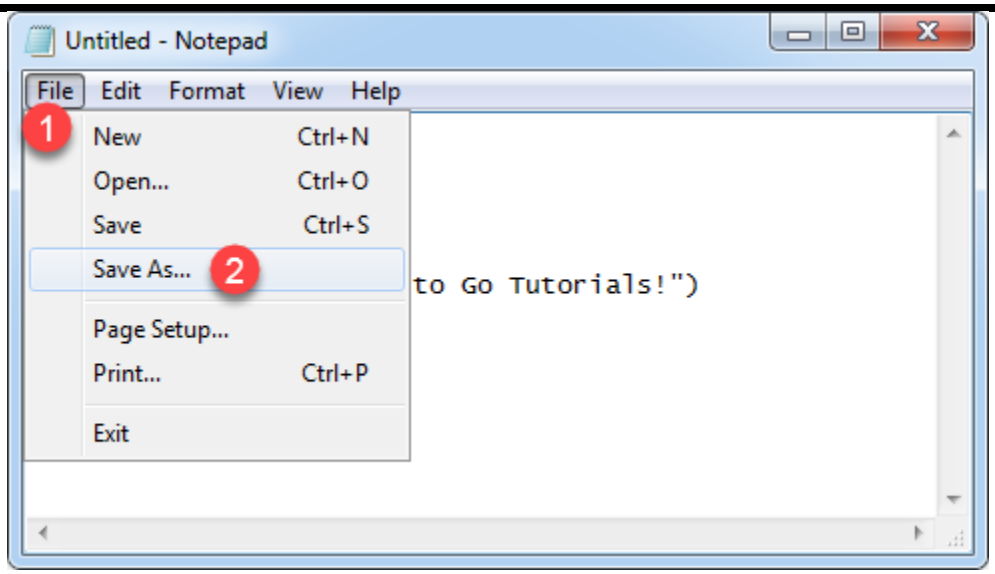


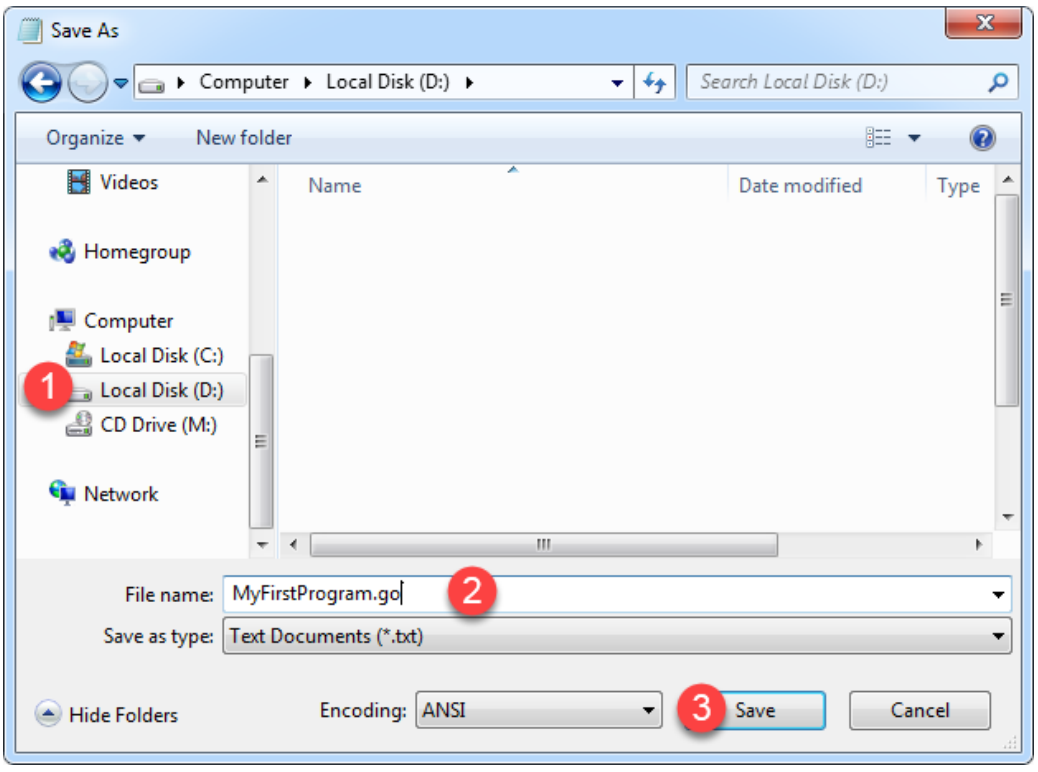
از پنجره باز شده به صورت زیر به سربرگ View رفته و تیک کنار گزینه Hide Extension for khnown file types را بردارید:

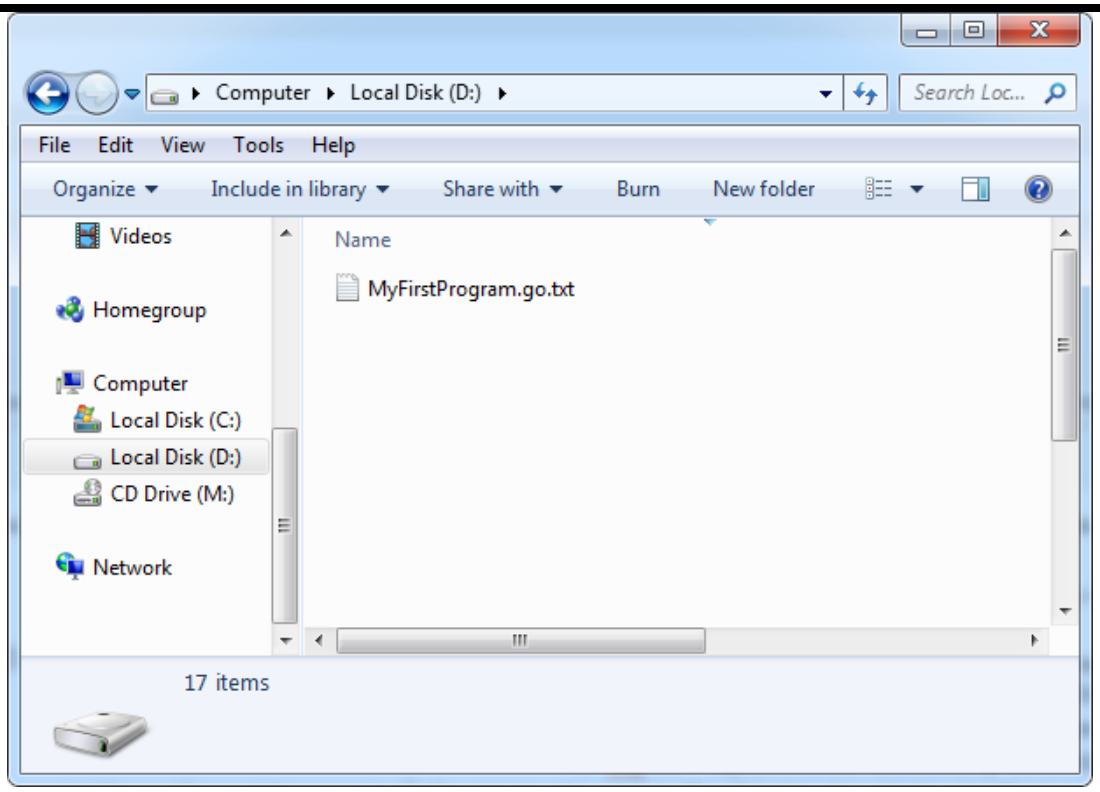


در ادامه شما را نحوه ایجاد اولین برنامه در Go را توضیح می‌دهیم. همانطور که گفته شد، شما برای کامپایل و اجرای برنامه‌های Go به کامپایلر این زبان نیاز دارید، که آن را در درس قبل نصب کردیم و الان فرض می‌کنیم که شما هیچ IDE یا محیط کدنویسی در اختیار ندارید و می‌خواهید یک برنامه Go بنویسید. در این برنامه می‌خواهیم پیغام Welcome to Go Tutorials چاپ شود. ابتدا یک ویرایشگر متن مانند Notepad را باز کرده و کدهای زیر را در داخل آن نوشته (حروف بزرگ و کوچک را رعایت کنید) و با پسوند go ذخیره کنید :

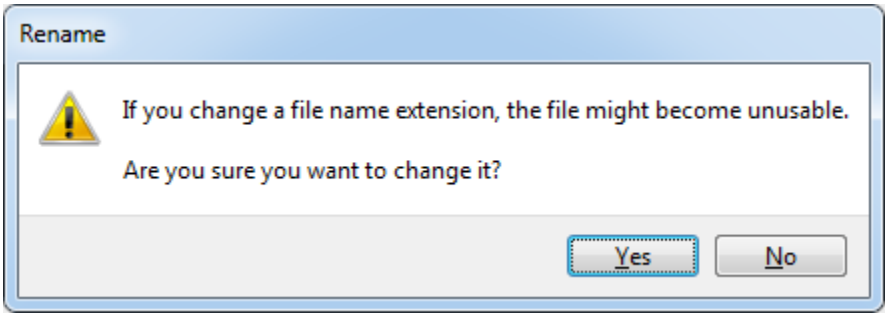
```go
package main

import "fmt"

func main(){
   fmt.Println("Welcome to Go Tutorials!")
}
```


Untitled - Notepad
File Edit Format View Help
1
New Ctrl+N
Open... Ctrl+O
Save Ctrl+S
Save As... 2
Page Setup...
Print... Ctrl+P
Exit
to Go Tutorials!")

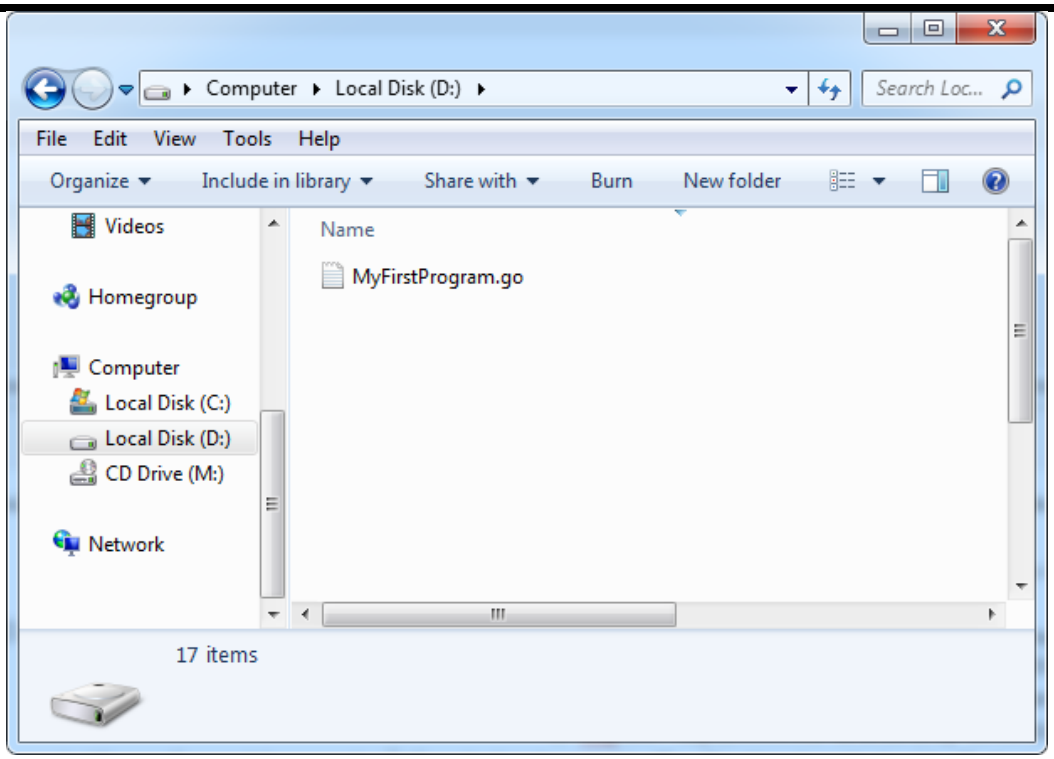

Save As
Computer ▸ Local Disk (D:) ▸
Search Local Disk (D:)
Organize
New folder
Videos
Homegroup
Computer
Local Disk (C:)
1
Local Disk (D:)
CD Drive (M:)
Network
Name
Date modified
Type
File name: MyFirstProgram.go 2
Save as type: Text Documents (*.txt)
Hide Folders
Encoding: ANSI
3 Save
Cancel



همانطور که مشاهده می‌کنید، بعد از ذخیره، فایل با پسوند `MyFirstProgram.go.txt` ذخیره می‌شود که شما باید پسوند `.txt` آن را حذف کنید. هنگام پاک کردن پسوند، پیغامی به صورت زیر ظاهر می‌شود که شما باید بر روی گزینه `Yes` کلیک کنید:



تا شکل نهایی فایل به صورت زیر در آید:



حال نوبت به اجرای برنامه می‌رسد. فایل ما در درایو D قرار دارد. ابتدا cmd را باز کرده و کد زیر را در داخل آن نوشته و دکمه Enter را می‌زنید :

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\siavash>d:

D:\>go run MyFirstProgram.go
Welcome to Go Tutorials!

D:\>
```

همانطور که در کد بالا مشاهده می‌کنید، برای اجرای کدهای Go ابتدا جمله go run و سپس نام پروژه به همراه پسوند آن را می‌نویسیم (مثل MyFirstProgram.go)

ساختار یک برنامه در Go

مثال بالا ساده‌ترین برنامه‌ای است که شما می‌توانید در Go بنویسید. هدف در مثال بالا نمایش یک پیغام در صفحه نمایش است. هر زبان برنامه نویسی دارای قواعدی برای کدنویسی است. اجازه بدهید هر خط کد را در مثال بالا توضیح بدهیم. در خط اول package

تعریف شده است که شامل کدهای نوشته شده توسط شما است و از تداخل نامها جلوگیری می‌کند. درباره package در درسهای آینده توضیح خواهیم داد. فقط این نکته را در همین ابتدا ذکر کنیم که وجود این خط برای اجرای کدها الزامی است.

Go دارای package هایی است که به صورت توکار و هنگام نصب کامپایلر Go نصب می‌شوند و هر کدام برای مقاصد خاصی مورد استفاده قرار می‌گیرند. یکی از این package ها، fmt می‌باشد. برای استفاده از این package ها از کلمه کلیدی import استفاده می‌کنیم (خط ۳). خط ۵ متد ()main یا متد اصلی نامیده می‌شود. هر متد شامل یک سری کد است که وقتی اجرا می‌شوند که متد را صدا بزنیم. درباره متد و نحوه صدا زدن آن در فصول بعدی توضیح خواهیم داد. متد ()main نقطه آغاز اجرای برنامه است. این بدان معناست که ابتدا تمام کدهای داخل متد ()main و سپس بقیه کدها اجرا می‌شود. درباره متد ()main در فصول بعدی توضیح خواهیم داد. متد ()main و سایر متدها دارای آکولاد و کدهایی در داخل آن‌ها می‌باشند و وقتی کدها اجرا می‌شوند که متدها را صدا بزنیم. مثالی از یک خط کد در Go به صورت زیر است :

```
fmt.Println("Welcome to Go Tutorials!")
```

در خط ۵ آکولاد ({) نوشته شده است. آکولاد برای تعریف یک بلوک کد به کار می‌رود. Go یک زبان ساخت یافته است که شامل کدهای زیاد و ساختارهای فراوانی می‌باشد. هر آکولاد باز ({) در Go باید دارای یک آکولاد بسته (}) نیز باشد. همه کدهای نوشته شده از خط ۵ تا خط ۷ یک بلوک کد است. این خط کد پیغام Welcome to Go Tutorials! را در صفحه نمایش نشان می‌دهد. از متد ()Println که در داخل package، fmt قرار دارد، برای چاپ یک رشته استفاده می‌شود. یک رشته گروهی از کاراکترها است که به وسیله دابل کوتیشن (") محصور شده است. مانند "Welcome to Visual Go Tutorials!":

یک کاراکتر می‌تواند یک حرف، عدد، علامت یا .... باشد. در کل مثال بالا نحوه استفاده از متد ()Println نشان داده شده است. این متد یک متد از پکیج fmt بوده و از آن برای چاپ مقادیر استفاده می‌شود. Go فضای خالی و خطوط جدید را نادیده می‌گیرد. همیشه به یاد داشته باشید که Go به بزرگی و کوچکی حروف حساس است. یعنی به طور مثال MAN و man در Go با هم فرق دارند. رشته‌ها و توضیحات از این قاعده مستثنی هستند که در درسهای آینده توضیح خواهیم داد. مثلاً کدهای زیر با خطا مواجه می‌شوند و اجرا نمی‌شوند :

```
fmt.println("Welcome to Go Tutorials!")
fmt.PRIintln("Welcome to Go Tutorials!")
FMT.Println("Welcome to Go Tutorials!")
```

تغییر در بزرگی و کوچکی حروف از اجرای کدها جلوگیری می‌کند. اما کد زیر کاملاً بدون خطا است :

```
fmt.Println("Welcome to Go Tutorials!")
```

همیشه کدهای خود را در داخل آکولاد بنویسید.

```
{
   statement1
}
```

این کار باعث می‌شود که کدنویسی شما بهتر به چشم بیاید و تشخیص خطاها راحت‌تر باشد. در زبان Go نیازی به سمیکالن (;) ندارید. همین که بعد از هر خط یک بار Enter بزنید به منزله این است که آن خط به پایان رسیده است. مثلاً دو خط زیر دو دستور جدا هستند:

```
fmt.Println("Welcome to Go Tutorials!")
fmt.Println("Welcome to Go Tutorials!")
```

توضیحات

وقتی که کدی تایپ می کنید شاید بخواهید که متنی جهت یادآوری وظیفه آن کد به آن اضافه کنید. در Go (و بیشتر زبانهای برنامه نویسی) می توان این کار را با استفاده از توضیحات انجام داد. توضیحات متونی هستند که توسط کامپایلر نادیده گرفته می شوند و به عنوان بخشی از کد محسوب نمی شوند.

هدف اصلی از ایجاد توضیحات، بالا بردن خوانایی و تشخیص نقش کدهای نوشته شده توسط شما، برای دیگران است. فرض کنید که می خواهید در مورد یک کد خاص، توضیح بدهید، می توانید توضیحات را در بالای کد یا کنار آن بنویسید. از توضیحات برای مستند سازی برنامه هم استفاده می شود. در برنامه زیر نقش توضیحات نشان داده شده است :

```
1  package main
2
3  import "fmt"
4
5  func main(){
6     //This line will print the message hello world
7     fmt.Println("Hello World!")
8  }
```

در کد بالا، خط ۶ کد بالا یک توضیح درباره خط ۷ است که به کاربر اعلام می کند که وظیفه خط ۷ چیست ؟ با اجرای کد بالا فقط جمله Hello World چاپ شده و خط اول در خروجی نمایش داده نمی شود چون کامپایلر توضیحات را نادیده می گیرد. همانطور که مشاهده می کنید برای درج توضیحات در Go از علامت // استفاده می شود. اگر توضیح درباره یک کد به بیش از یک خط نیاز باشد از توضیحات

چند خطی استفاده می‌شود. توضیحات چند خطی با */ شروع و با /* پایان می‌یابند. هر نوشته‌ای که بین این دو علامت قرار بگیرد جز توضیحات محسوب می‌شود.

```
package main

import "fmt"

func main(){
   /*This line will print
   the message hello world*/
   fmt.Println("Hello World!")
}
```

کاراکترهای کنترلی

کاراکترهای کنترلی کاراکترهای ترکیبی هستند که با یک بک اسلش (\) شروع می‌شوند و به دنبال آنها یک حرف یا عدد می‌آید و یک رشته را با فرمت خاص نمایش می‌دهند. برای مثال برای ایجاد یک خط جدید و قرار دادن رشته در آن می‌توان از کاراکتر کنترلی \n استفاده کرد :

```
fmt.Println("Hello\nWorld!")
```

```
Hello
World
```

مشاهده کردید که کامپایلر بعد از مواجهه با کاراکتر کنترلی \n نشانگر ماوس را به خط بعد برده و بقیه رشته را در خط بعد نمایش می‌دهد. متد Println() هم مانند کاراکتر کنترلی \n یک خط جدید ایجاد می‌کند، البته بدین صورت که در انتهای رشته یک کاراکتر کنترلی \n اضافه می‌کند :

```
fmt.Println("Hello World!")
```

کد بالا و کد زیر هیچ فرقی با هم ندارند :

```
fmt.Print("Hello World!\n")
```

متد Print() کارکردی شبیه به Println() دارد با این تفاوت که نشانگر ماوس را در همان خط نگه می‌دارد و خط جدید ایجاد نمی‌کند.

جدول زیر لیست کاراکترهای کنترلی و کارکرد آنها را نشان می‌دهد :

عملکرد	کاراکتر کنترلی	عملکرد	کاراکتر کنترلی
Form Feed	\f	چاپ کوتیشن	\’
خط جدید	\n	چاپ دابل کوتیشن	\”
سر سطر رفتن	\r	چاپ بک اسلش	\\
حرکت به صورت افقی	\t	چاپ فضای خالی	\0
حرکت به صورت عمودی	\v	صدای بیپ	\a
چاپ کاراکتر یونیکد	\u	حرکت به عقب	\b

ما برای استفاده از کاراکترهای کنترلی از بک اسلش (\) استفاده می‌کنیم. از آنجاییکه \ معنای خاصی به رشته‌ها می‌دهد برای چاپ بک اسلش (\) باید از (\\) استفاده کنیم :

```
fmt.Println("We can print a \\ by using the \\\\ escape sequence.")
```

```
We can print a \ by using the \\ escape sequence.
```

یکی از موارد استفاده از \\، نشان دادن مسیر یک فایل در ویندوز است :

```
fmt.Println("C:\\Program Files\\Some Directory\\SomeFile.txt")
```

```
C:\Program Files\Some Directory\SomeFile.txt
```

از آنجاییکه از دابل کوتیشن (“) برای نشان دادن رشته‌ها استفاده می‌کنیم برای چاپ آن از \” استفاده می‌کنیم :

```
fmt.Println("I said, \"Motivate yourself!\".")
```

```
I said, "Motivate yourself!".
```

همچنین برای چاپ کوتیشن (‘) از \’ استفاده می‌کنیم :

```
fmt.Println("The programmer\'s heaven.")
```

```
The programmer's heaven.
```

برای ایجاد فاصله بین حروف یا کلمات از \t استفاده می‌شود :

```
fmt.Println("Left\tRight")
```

```
Left    Right
```

هر تعداد کاراکتر که بعد از کاراکتر کنترلی \r۳ بیایند به اول سطر منتقل و جایگزین کاراکترهای موجود می‌شوند :

```
fmt.Println("Mitten\rK")
```

```
K
```

مثلاً در مثال بالا کاراکتر K بعد از کاراکتر کنترلی \r۳ آمده است. کاراکتر کنترلی حرف K را به ابتدای سطر برده و جایگزین حرف M می‌کند.

برای چاپ کاراکترهای یونیکد می‌توان از \u استفاده کرد. برای استفاده از \u، مقدار در مبنای ۱۶ کاراکتر را درست بعد از علامت \u قرار می‌دهیم. برای مثال اگر بخواهیم علامت کپی رایت (©) را چاپ کنیم، باید بعد از علامت \u مقدار A900 را قرار دهیم مانند :

```
fmt.Println("\u00A9")
```

```
©
```

برای مشاهده لیست مقادیر مبنای ۱۶ برای کاراکترهای یونیکد به لینک زیر مراجعه نمایید :

http://www.ascii.cl/htmlcodes.htm

اگر کامپایلر به یک کاراکتر کنترلی غیر مجاز برخورد کند، برنامه پیغام خطا می‌دهد. بیشترین خطا زمانی اتفاق می افتد که برنامه نویس برای چاپ اسلش (\) از \\ استفاده می‌کند. برای دریافت اطلاعات بیشتر در مورد کاراکترهای کنترلی به لینک زیر مراجعه کنید :

https://msdn.microsoft.com/en-us/library/h21280bw.aspx

متغیر

متغیر مکانی از حافظه است که شما می‌توانید مقادیری را در آن ذخیره کنید. می‌توان آن را به عنوان یک ظرف تصور کرد که داده‌های خود را در آن قرار داده‌اید. محتویات این ظرف می‌تواند پاک شود یا تغییر کند. هر متغیر دارای یک نام نیز هست. که از طریق آن می‌توان متغیر را از دیگر متغیرها تشخیص داد و به مقدار آن دسترسی پیدا کرد. همچنین دارای یک مقدار می‌باشد که می‌تواند توسط کاربر انتخاب شده باشد یا نتیجه یک محاسبه باشد. مقدار متغیر می‌تواند تهی نیز باشد. متغیر دارای نوع نیز هست بدین معنی که نوع آن با نوع داده‌ای که در آن ذخیره می‌شود یکی است.

متغیر دارای عمر نیز هست که از روی آن می‌توان تشخیص داد که متغیر باید چقدر در طول برنامه مورد استفاده قرار گیرد. و در نهایت متغیر دارای محدوده استفاده نیز هست که به شما می‌گوید که متغیر در چه جای برنامه برای شما قابل دسترسی است. ما از متغیرها به عنوان یک انبار موقتی برای ذخیره داده استفاده می‌کنیم. هنگامی که یک برنامه ایجاد می‌کنیم احتیاج به یک مکان برای ذخیره داده، مقادیر یا داده‌هایی که توسط کاربر وارد می‌شوند، داریم. این مکان، همان متغیر است.

برای این از کلمه متغیر استفاده می‌شود چون ما می‌توانیم بسته به نوع شرایط هر جا که لازم باشد، مقدار آن را تغییر دهیم. متغیرها موقتی هستند و فقط موقعی مورد استفاده قرار می‌گیرند که برنامه در حال اجراست و وقتی شما برنامه را می‌بندید محتویات متغیرها نیز پاک می‌شود. قبلاً ذکر شد که به وسیله نام متغیر می‌توان به آن دسترسی پیدا کرد. برای نامگذاری متغیرها باید قوانین زیر را رعایت کرد :

- **نام متغیر باید با یکی از حروف الفبا** `(a-z or A-Z)` **یا علامت _ شروع شود.**
- **نمی‌تواند شامل کاراکترهای غیرمجاز مانند** `., $, ^, ?, #` **باشد.**
- **نمی‌توان از کلمات رزرو شده در** `Go` **برای نام متغیر استفاده کرد.**
- **نام متغیر نباید دارای فضای خالی** `(spaces)` **باشد.**
- **اسامی متغیرها نسبت به بزرگی و کوچکی حروف حساس هستند. در** `Go` **دو حرف مانند** `a` **و** `A` **دو کاراکتر مختلف به حساب می‌آیند.**

دو متغیر با نامهای `myNumber` و `MyNumber` دو متغیر مختلف محسوب می‌شوند چون یکی از آنها با حرف کوچک `m` و دیگری با حرف بزرگ `M` شروع می‌شود. متغیر دارای نوع هست که نوع داده‌ای را که در خود ذخیره می‌کند را نشان می‌دهد. در درس بعد در مورد انواع داده ها در Go توضیح می دهیم. لیست کلمات کلیدی Go، که نباید از آنها در نامگذاری متغیرها استفاده کرد در زیر آمده است: