**متغیرها**

بیشتر اوقات برنامه‌های جاوا اسکریپت نیاز دارند تا با اطلاعات کار کنند. بطور نمونه :

1 یک فروشگاه آنلاین - اطلاعات شامل کالاهایی که فروخته می‌شوند و سبد خرید بشود.

2 یک نرم‌افزار چت - اطلاعات شامل کاربران، پیام‌ها و دیگر موارد می‌شود.

متغیرها برای ذخیره سازی اطلاعات استفاده می‌شوند.

**متغیر**

یک متغیر در واقع فضایی برای ذخیره سازی اطلاعات با نامی مشخص است. ما از متغیرها برای ذخیره سازی نام کالاها، بازدید‌کنندگان و دیگر انواع اطلاعات استفاده می‌کنیم.

برای ساخت یک متغیر در جاوا اسکریپت از let استفاده کنید.

عبارت زیر یک متغیر با نام message تعریف میکند. به این عمل define یا declare گفته می‌شود.

کد

حالا می‌توانیم اطلاعاتی را از طریق عملگر = در این متغیر قرار دهیم.

کد

حال رشته کاراکترهایی که نوشتیم در بخشی از حافظه کامپیوتر که در اختیار این متغیر می‌باشد ذخیره شده است.

کد

به منظور احتصار می‌توانیم تعریف متغیر و مقداردهی به آن را در یک خط قرار دهیم.

کد

همینطور می‌توانیم چند متغیر را در یک خط تعریف کنیم.

کد

این مورد ممکن است کوتاه‌تر به نظر آید اما ما پیشنهاد نمی‌کنیم. چراکه خوانایی را پایین می‌آورد. لطفا برای تعریف متغیرها از خطوط جداگانه استفاده کنید.

کد

بعضی برنامه‌نویسان متغیرهای متعدد را به این صورت در چند خط می‌نویسند :

کد

همینطور برخی به صورت comma-first اینکار را انجام می‌دهند :

کد

از لحاظ فنی همه این حالات کار خواهند کرد و انتخاب آن‌ها به سلایق و زیبایی شناسی شخصی مربوط است.

**Var بجای let**

در اسکریپت‌های قدیمی ممکن است بجای کلیدواژه let با کلیدواژه var برخورد کنید.

کد

کلیدواژه var تقریبا با let یکسان است و یک متغیر را تعریف می‌کند، ولی روش کار آن قدیمی است.

تفاوت کوچکی بین این دو وجود دارد که در حال حاضر به آن نمی‌پردازیم و در مبحث The old "var"به آن خواهیم پرداخت.

**یک مقایسه‌ی عینی \*\*\***

برای درک مفهوم متغیر، می‌توانیم آن را یک جعبه برای نگهداری اطلاعات تصور کنیم، که یک نام منحصربفرد روی آن چسبانده‌ایم.

برای نمونه، تصور کنید متغیر message در جعبه‌ای با برچسب message و با مقدار Hello! باشد :

تصویر

ما می‌توانیم هر مقداری در آن قرار دهیم. همینطور می‌توانیم هر چند بار که خواستیم این مقدار با تغییر دهیم.

کد

وقتی مقدار تغییر کرد، اطلاعات قبلی از آن پاک خواهند شد :

تصویر

ما همینطور می‌توانیم دو متغیر تعریف کرده و اطلاعات یکی را در دیگری کپی کنیم.

کد

**زبان‌های Functional**

جالب است بدانید زبان‌های Functional مانند Scala یا Erlang تغییر مقدار متغیر را ممنوع کرده‌اند.

در این نوع زبان‌ها وقتی مقداری را در جعبه قرار می‌دهیم تا ابد آنجا می‌ماند و اگر بخواهیم مقداری دیگر را ذخیره کنیم، ما را مجبور می‌کنند تا جعبه‌ای جدید بسازیم.

اگرچه در نگاه اول این زبان‌ها عجیب به نظر می‌رسند، ولی کاملا توانایی توسعه‌ (در پروژه‌های) جدی را دارند. مهم‌تر از آن در زمینه‌هایی مانند محاسبات موازی (Parallel Computation) این محدودیت‌ها تبدیل به مزیت می‌شوند. مطالعه‌ی چنین زبان‌هایی (حتی اگر به این زودی قصد استفاده از آنها را ندارید) برای وسیع شدن دیدتان، توصیه می‌شود.

**نام‌گذاری متغیرها**

دو محدودیت برای نام‌گذاری متغیرها در جاوا اسکریپت وجود دارد :

1 نام باید فقط از کاراکترها، اعداد یا علامت $ و \_ تشکیل شده باشد.

2 اولین کاراکتر نمی‌تواند عدد باشد.

بطور نمونه :

کد

زمانی‌که به یک نام متشکل از چند کلمه است، عموما از روش camelCase استفاده می‌شود. به این صورت که کلمات یکی پس از دیگری پشت هم نوشته می‌شوند و حرف اول هر کلمه کاراکتر بزرگ هست : myVeryLongName .

همینطور از $ و \_ نیز می‌توان استفاده نمود. معنی خاصی ندارند و همانند حروف قابل استفاده هستند.

این نام‌ها معتبر هستند :

کد

این نام‌ها نامعتبر هستند :

کد

**بزرگی و کوچکی حروف مهم است**

متغیر apple با Apple متفاوت است.

**کاراکترهای غیر انگلیسی مجاز هستند ولی توصیه نمی‌شوند**

امکان استفاده از کاراکترهای زبان‌های دیگر وجود دارد. مانند :

کد

از نظر فنی اینها درست کار می‌کنند ولی بر اساس یک رسم بین‌المللی برای نام متغیرها از زبان انگلیسی استفاده می‌شود. حتی اگر اسکریپتی کوچک می‌نویسیم، ممکن است تا مدت طولانی‌ای مورد استفاده و توسعه قرار بگیرد و اشخاصی از سایر کشورها ممکن است نیاز باشد روزی آن اسکریپت را بخوانند.

**نام‌های رِزِرو شده**

تعدادی نام رزرو شده وجود دارد که نمی‌توان از آنها برای نام متغیرها استفاده نمود چراکه آنها توسط خودِ زان جاوا اسکریپت استفاده شده‌اند.

برای نمونه let، class، return و function رزرو هستند.

این کد ارور Syntax خواهد داشت :

کد

**مقداردهی بدون use strict**

در گذشته این امکان وجود داشت تا متغیری را بدون استفاده از let و صرفا با مقداردهی تعریف نمود. درصورتیکه از use strict استفاده نکنیم، این امکان همچنان وجود دارد،

کد

این روش مناسب نیست و در صورت وجود use strict با خطا مواجه خواهد شد.

کد

**ثابت‌ها**

برای تعریف ثابت (متغیری که تغیر نمی‌کند) از const بجای let استفاده می‌کنیم :

کد

متغیرهایی که با const ساخته می‌شود را ثابت (constant) می‌گوییم. آنها قابل تغییر نیستند و تلاش برای اینکار با خطا روبرو خواهد شد :

کد

وقتی برنامه‎‌نویس اطمینان دارد که متغیری هیچگاه تغییر نمی‌کندُ می‌تواند آن را به عنوان const تعریف کند تا برای دیگران نیز این موضوع واضح باشد.

**ثابت‌های با حروف بزرگ**

به صورت یک عُرفِ همه‌گیر، از ثابت‌های با حروف بزرگ به عنوان نام مستعار برای مقادیری که به خاطر سپردن آنها دشوار است، استفاده می‌شود. این دسته از ثابت‌ها اصطلاحا prior to execution (پیش از اجرای برنامه) مقدارشان قطعی است.

در ثابت‌ها از حروف بزرگ و \_ استفاده می‌شود :

کد

مزایا :

COLOR\_ORANGE بسیار راحت‌تر از #FF7F00 به یاد می‌ماند.

اشتباه در نوشتن #FF7F00 بسیار محتمل‌تر از COLOR\_ORANGE است.

در هنگام خواندن کد، خواندن COLOR\_ORANGE بسیار معنادارتر از #FF7F00 است.

چه زمانی باید از حروف بزرگ و چه زمانی باید از حروف معمولی برای نام‌گذاری یک constant استفاده کنیم؟

Constant بودن یک متغیر صرفا بدین معناست که مقدار آن تغییر نخواهد کرد. یک دسته از ثابت‌ها پیش از اجرای برنامه مقدارشان قطعی خواهد بود و دسته دگیر در حین اجرای (Run Time) برنامه مقدارشان مشخص می‌شود (و دیگر مقدارشان تغییر نمی‌کند)

برای نمونه :

کد

مقدار ثابت pageLoadTime پیش از اجرای برنامه مشخص نبوده و به همین دلیل به صورت عادی نوشته شده است. اما همچنان یک ثابت است چراکه زمان اجرای برنامه بعد دیگر تغییر نخواهد کرد.

به تعبیری دیگر ثابت‌های با حروف بزرگ زمانی که اصطلاحا مقداری به صورت hard code شده داریم، استفاده می‌شود (مانند کد رنگ قرمز).

**نام‌های مناسب انتخاب کنید**

یک مورد مهم دیگر در مورد متغیرها وجود دارد.

لطفا نام متعیرهای خود را با حساسیت انتخاب کنید و برای اینکار خوب فکر کنید.

انتخاب نام برای متغیرها یکی از کارهای مهم و پیچیده در برنامه‌نویسی است. یک نگاه سریع به نام متغیرها می‌تواند تفاوت یک برنامه‌نویس تازه‌کار و با تجربه را نشان دهد.

در پروژه‌های واقعی، بجای از صفر نوشتن برنامه‌ها، بیشتر زمان صرف اصلاح و توسعه کدهای موجود می‌شود. وقتی پس از مدتی به کدهای قبلی بازمی‌گردیم، بدست آوردن اطلاعات از آن دسته کدهایی که نام‌گذاری‌هیا خوبی دارند بسیار راحت‌تر است (منظور متغیرهایی است که نام‌های خوبی دارند).

لطفا پیش از انتخاب نام برای یک متغیر خوب به آن فکر کنید. ثمره‌ی آن را خواهید دید.

بعضی از دستورالعمل‌های مفید :

۱ از نام‌هایی که برای انسان قابل فهم است استفاده کنید مانند userName یا shoppingCart.

۲ از نام‌های مخفف یا کوتاه استفاده نکنید مانند a، b، c، مگر آنکه بدانید چه می‌کنید.

3 نام‌های کاملا واضح و مختصر انتخاب کنید. نمونه‌هایی از نام‌های بد data و value هستند. این نام‌ها هیچ‌چیز را توضیح نمی‌دهند. استفاده از این دست نام‌ها فقط زمانی قابل قبول است که محتوای کدی که می‌نویسید استثنا مشخص کند که data یا value به چه چیزی اشاره می‌کند.

4 با ذهن خود و تیم کاریتان توافق نظر داشته باشید. اگر به مخاطبان وب‌سایت‌تان user می‌گویید، متغیرهای مرتبط را بجای currentVisitor یا newManInTown باید currentUser یا newUser بنامید.

**ایجاد یا استفاده مجدد**

به عنوان آخرین نکته برخی برنامه‌نویسان تنبل بجای تعریف متغیرهای جدید، از متغیرهای موجود دوباره استفاده می‌کنند.

در نتیجه متغیرهای آن‌ها مانند جعبه‌ایست که افراد متفاوت چیزهای مختلفی درون آن ریخته‌اند و نام روی برچسب آن را تغییر نداده‌اند.

این دسته از برنامه‌نویسان در تعریف متغیر مقداری صرفه‌جویی، ولی مدت زمان زیادی از زمان رفع خطاها می‌افزایند.

بهینه‌سازی‌هایی که در مرورگرهای امروزی و همینطور کم‌حجم‌کننده‌های جاوا اسکریپت صورت می‌گیرد، در بوجود نیامدن مشکلات عملکردی (Performance) برنامه‌های ما کمک می‌کنند. حتی استفاده از متغیرهای مختلف برای مقادیر مختلف به موتور جاوا اسکریپت کمک می‌کند تا کد شما را بهینه کند.