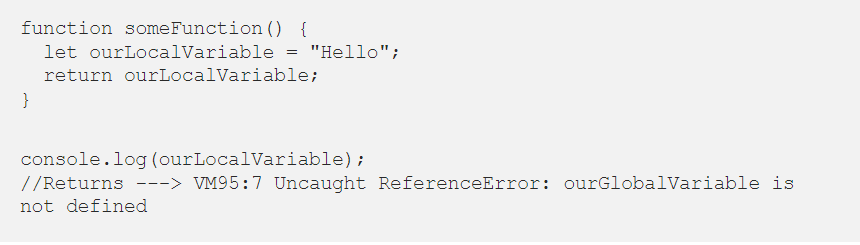
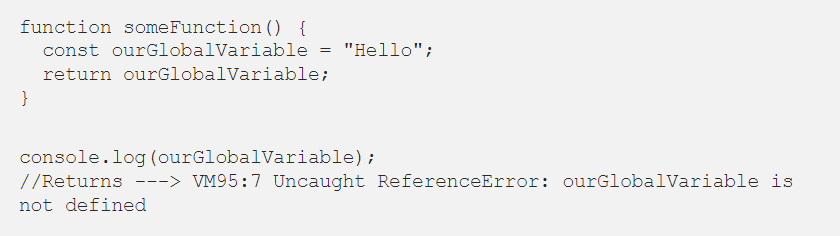
**(Function Scope)**

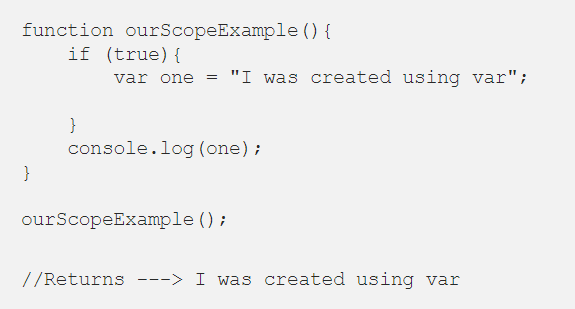
* JavaScript-ում յուրաքանչյուր ֆունկցիա ստեղծում է նոր scope:
* Var, let և const-ով հայտարարված փոփոխականները միանգամայն նման են ֆունկցիայի ներսում հայտարարվելիս:
* JavaScript ֆունկցիայի մեջ հայտարարված փոփոխականները դառնում են LOCAL այդ ֆունկցիայի համար:
* Function Scope-ում հայտարարված փոփոխականները դառնում են անհասանելի և հեռացվում են, երբ ավարտվում է ֆունկցիայի աշխատանքը, և պահպանվում են հիշուղությունում։ Այդ պատճառով էլ այդ փոփոխականները դառնում են հասանելի, երբ ֆունկցիան կանչվում է և դառնում են անհասանելի երբ ավարտվում է ֆունկցիայի աշխատանքը։
* Ֆունկցիայի ներսում հայտարարված փոփոխականը հասանելի չէ ֆունկցիայի սկոպից դուրս։

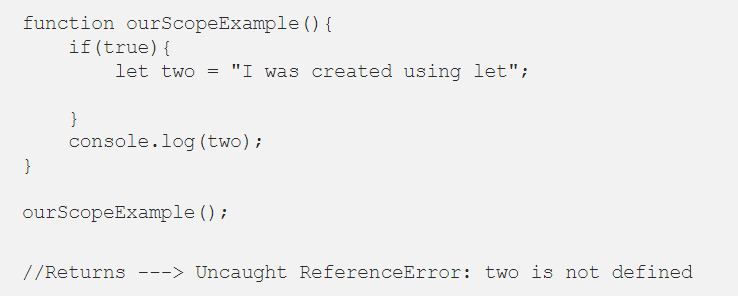


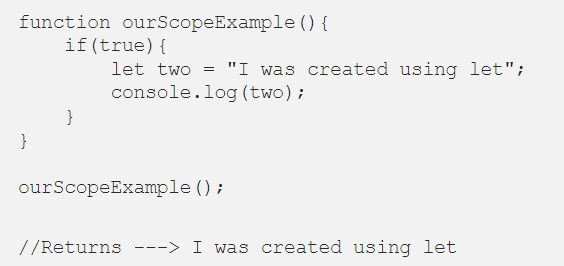
****

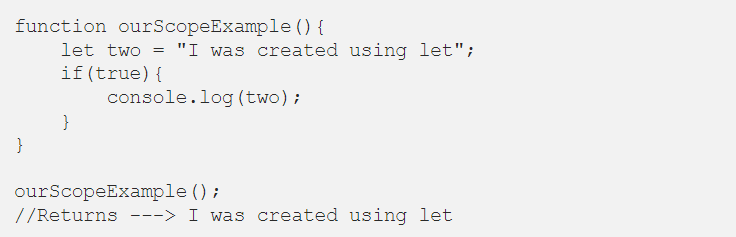
**(Block Scope)**

* Մինչև ES6-ը (2015), JavaScript-ն ուներ միայն Global Scope և Function Scope: ES6-ը ներկայացրեց երկու կարևոր JavaScript հիմնաբառեր՝ let և const: Այս երկու հիմնաբառերը ապահովում են Block Scope JavaScript-ում:
* { } բլոկի ներսում հայտարարված փոփոխականները չեն կարող մուտք գործել բլոկից դուրս.
* Var-ով հայտարարված փոփոխականները **չեն** կարող ունենալ Block Scope:

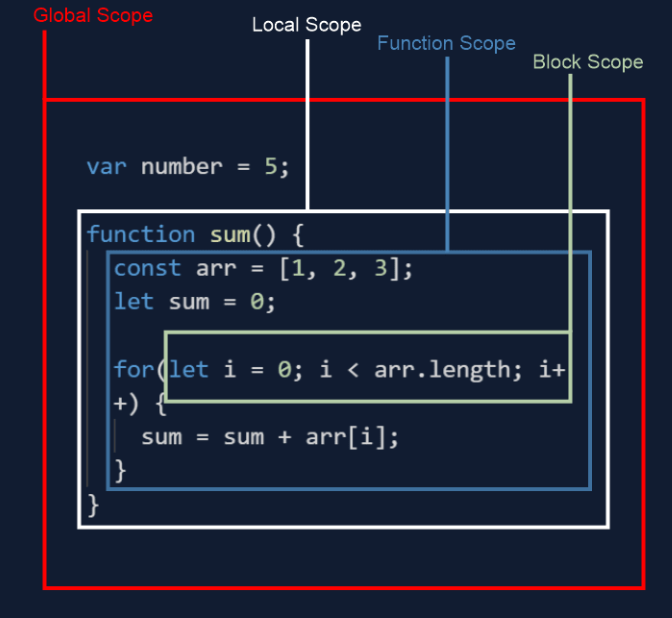
****

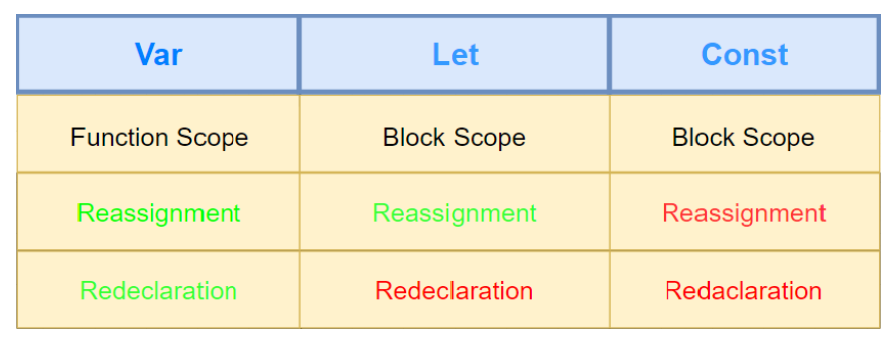
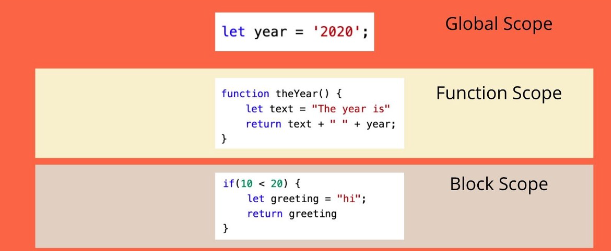
****

****

****

**(Function and Block Scopes)**

****

****