Praktikat e mira te kodimit

Hyrje

Teknikat e mira të kodimit dhe praktikat e programimit janë shenjat dalluese të një programues profesionist. Pjesa më e madhe e programit konsiston në marrjen e një numri të madh të zgjedhjesh të vogla të problemit, kështu përpiqesh për të zgjidhur një sërë më të madhe të problemeve. Sa mençurisht këto zgjedhje janë bërë varet në masë të madhe me aftësitë dhe ekspertizën e programuesit.

Ky kapitull trajton disa teknika themelore të kodimit dhe ofron një koleksion të praktikave të kodimit nga e cila kemi plot për të mësuar. Teknikat e kodimit janë kryesisht ato që përmirësojnë leximin dhe mirëmbajtjen e kodit, ndërsa praktikat e programimit janë kryesisht enhancements të performancës.

Lexueshmëri e kodit burim ka një ndikim të drejtpërdrejtë në se sa mirë një zhvillues e kupton një sistem kompjuterik. Kodi maintainability i referohet sa lehtë që sistemi software mund të ndryshohet për të shtuar veçori të reja, të ndryshuar tiparet ekzistuese, fix bugs, ose të përmirësuar punën. Edhe pse lexueshmëri dhe maintainability janë rezultat i shumë faktorëve, një aspekt të veçantë të zhvillimit të softuerit në të cilin të gjithë zhvilluesve të ketë një ndikim është coding teknikë. Metoda më e lehtë për të siguruar që një ekip i zhvilluesve do të japin kodin e cilësisë është të krijojë një standard kodim, e cila është e detyrueshme pastaj në komente rutinë kodit.

Standardet e kodimit dhe shqyrtimet e kodit

Një standard i plotë kodim përfshin të gjitha aspektet e ndërtimit të kodit dhe, ndërsa zhvilluesit duhet të ushtrojnë kujdes në zbatimin e saj, ajo duhet të ndiqet nga afër. Kodi burimor i plotësuar duhet të pasqyrojnë një stil të harmonizuar, si në qoftë se një zhvillues i vetëm ka shkruar kodin në një seancë. Në fillim të një projekti software, të krijojë një standard kodim për të siguruar që të gjithë zhvilluesit për këtë projekt janë duke punuar në bashkëpunim. Kur projekti software do të përfshijnë kod burim ekzistues, ose kur kryejnë mirëmbajtje mbi një sistem ekzistues software, standardi coding duhet të deklarojë si të merren me bazë ekzistuese të kodit.

Megjithëse qëllimi kryesor për kryerjen e rishikimeve kod përgjatë ciklit jetësor të zhvillimit është të identifikojë defektet në kodin, komente mund të përdoret gjithashtu për të zbatuar standardet e kodimit në mënyrë uniforme. Aderimi në një standard coding mund të jetë e mundshme vetëm kur ndiqen gjatë gjithë projektit software nga fillimi në përfundimin. Kjo nuk është praktike, as nuk është e kujdesshme, për të vendosur një standard të kodues pas faktit.

Teknikat e kodimit

Teknikat e kodimit përfshijnë shumë aspekte të zhvillimit të softuerit dhe, edhe pse ata zakonisht nuk kanë asnjë ndikim në funksionimin e aplikimit, ato kontribuojnë në një të kuptuar të mirë të kodit burim. Për qëllimet e këtij dokumenti, të gjitha format e kodit burimor janë konsideruar, duke përfshirë programimin, scripting, Markup, dhe gjuhë query.

Teknikat e coding përcaktuara këtu nuk janë propozuar për të formuar një grup të papërkulur të standardeve coding. Përkundrazi, ata kanë për qëllim të shërbejë si një udhërrëfyes për zhvillimin e një standard kodim për një projekt të veçantë software.

Teknikat coding janë të ndarë në tre seksione:

* [Emrat](https://msdn.microsoft.com/en-us/library/aa260844(v=vs.60).aspx#cfr_names)
* [Koment](https://msdn.microsoft.com/en-us/library/aa260844(v=vs.60).aspx#cfr_comments)et
* [Format](https://msdn.microsoft.com/en-us/library/aa260844(v=vs.60).aspx#cfr_format)i

Emrat

Ndoshta një nga ndihmat më me ndikim për të kuptuar rrjedhën logjike të një kërkese është se si quhen elementet e ndryshme të aplikimit. Një emër duhet të them se "çfarë" dhe jo "si." Duke shmangur emrat që ekspozojnë zbatimin themelor, i cili mund të ndryshojë, të ruajnë një shtresë e abstraksionit që thjeshton kompleksitetin. Për shembull, ju mund të përdorni GetNextStudent() në vend të GetNextArrayElement().

Një parim i emërtimit është se vështirësi në zgjedhjen e një emri të duhur mund të tregojnë se ju keni nevojë për të analizuar më tej ose të përcaktojë qëllimin e një artikulli. Bëni emrat kohë të mjaftueshme për të qenë kuptimplotë, por të shkurtër të mjaftueshme për të shmangur duke u gojor.Programuar, një emër unik shërben vetëm për të dallojnë një objekt nga një tjetër. emrat ekspresive të funksionojë si një ndihmë për lexuesit e njeriut;për këtë arsye, kjo ka kuptim për të siguruar një emër që lexuesi i njeriut mund ta kuptojë. Megjithatë, të jenë të sigurt se emrat e zgjedhura janë në përputhje me rregullat dhe standardet e gjuhës të zbatueshme s.

Në vijim janë rekomanduar emërtimin teknika:

Rutinat

Shmangni emrat e pakapshme që janë të hapura për interpretim subjektiv, të tilla si Analyze()një rutinë, ose xxK8për një ndryshore. Emra të tillë të kontribuojë në dykuptimësi më shumë se abstraksionit.

Në gjuhë object-oriented, ai është i tepërt për të përfshirë emra të klasës në emër të pronave të klasës, të tilla si Book.BookTitle. Në vend të kësaj, përdorni Book.Title.

Përdorni metodën folje-emër për emërtimin rutinat që kryejnë disa operacion në një objekt të caktuar, të tilla si CalculateInvoiceTotal().

Në gjuhë që lejojnë funksion mbingarkesat, të gjitha mbingarkesa duhet të kryejë një funksion të ngjashëm. Për ato gjuhë që nuk lejojnë funksion mbingarkesat, të krijojë një standard emërtimin që lidhet funksione të ngjashme.

Variablat //nga ketu fillon pjesa e re

Është mirë të vendoset një mbaresë dalluese (si mesatare, Sum, Min, Max, Index) në fund të një emri variable kur duket e përshtatshme.

Është mirë të përdoren palët e kundërta në emrat e variablave, si min / max, fillim / fund, dhe hap / mbyll.

Duhe qënë se shumica e emrave janë ndërtuar nga bashkimi i disa fjalëve së bashku, përdorni format të përzier të shkronjave të mëdha dhe të vogla për të lehtësuar leximin e tyre.Përveç kësaj, për të ndihmuar dallimin midis variablave, përdorni modelin Pascal ( ActivateScene() *marrë nga aplikacioni “Bazat e network-ut”*) për emrat e zakonshëm ku shkronjën e parë të çdo fjalë është kapitalizuar. Emrat e variablave, përdorni deve zorrë (stepNum *marrë nga aplikacioni “Bazat e network-ut”*) ku shkronja e parë të çdo fjalë përveç të parës është e kapitalizuar.

Variablat buleane duhet të përmbajë “Is” që nënkupton “PO/JO” ose “True/False”.Psh vlerat të tilla si “IsActive()” (*marrë nga aplikacioni “Bazat e network-ut”*)

Duhet shmangur përdorimi i termave të tilla si Flag/Shenjë/statusi kur kemi emërtimin e variablave të gjendjes, të cilat ndryshojnë nga variablat Boolean se ata mund të kenë më shumë se dy vlera të mundshme. Në vend të documentStatus, përdorin një emër më përshkrues të tilla si documentFormatType.

Edhe për një ndryshore jetëshkurtër që mund të shfaqen në vetëm disa rreshta të kodit, sërisht është mirë të përdoret një emër kuptimplotë. Përdoren emra me një shkronjë të variablave “i” apo ”j”, vetëm në raste të tilla ciklet e shkurtra for.

Nëse përdoret Konvencioni i emërtimit hungarez i Charles Simonyi, ose ndonjë derivat të tij, të zhvillohet një listë e prefikseve standarde për projektin për të ndihmuar zhvilluesit vazhdimisht me emërtimin e variablave.

Për emrat e variablave, ndonjëherë është e dobishme për të përfshirë ndonjë simbol që tregon qëllimin e variablit, të tilla si prefixing një g\_për variablat globale.

Konstantet duhet të jetë mbi të gjitha me shkronja të mëdha me vijë poshtë mes fjalëve, të tilla si NUM\_DAYS\_IN\_WEEK. Gjithashtu, fillojnë grupet e enumeracioneve me një prefiks të përbashkët, si FONT\_ARIAL dhe FONT\_ROMAN.

Të tjera

Duhet të kemi minimizuar përdorimin e shkurtesave. Në qoftë se shkurtimet janë përdorur, të jenë në përputhje me përdorimin e tyre. Një shkurtim duhet të ketë vetëm një kuptim dhe po ashtu, çdo fjalë shkurtuar duhet të ketë vetëm një shkurtim. Për shembull, në qoftë se duke përdorur *min* për të shkurtuar *minimum* , ta bëjmë këtë kudo dhe më vonë nuk e përdorim atë për të shkurtuar “*minutë”* .

Kur emërtimin e funksioneve, të përfshijë një përshkrim të vlerës duke u kthyer, të tilla si getValueLabel()(stepNum *marrë nga aplikacioni “Bazat e network-ut”*).

Emrat e file-ve dhe skedarëve/dosjeve, si emrat e procedurës, duhet të përshkruajnë me saktësi çfarë qëllimi ata shërbejnë.

Shmangni emrat ripërdorimin për elemente të ndryshme, të tilla si një emër rutinë i quajtur ProcessSales()dhe një ndryshore të quajtur iProcessSales.

Shmangni homonyms kur emërtimin elementet për të parandaluar konfuzionin gjatë shqyrtimeve të kodit, të tilla si *“write”* dhe *“right”* .

Kur emërtimin elemente, për të shmangur duke përdorur fjalë që zakonisht keqshqiptohen ose shqiptohen me vështirësi. Gjithashtu, të jenë të vetëdijshëm për dallimet që ekzistojnë mes anglishtes amerikane dhe britanike, të tilla si “color/colour” dhe “check/cheque”. Por dhe fjalë të tjera që shpesh shkruhen gabim. Kjo sepse do të ketë vështirësi në kërkim nëpër kod dhe problemi shtohet edhe më shumë kur ka më shumë se një zhvillues.

Shmangni përdorimin shenjat tipografike për të identifikuar llojet e të dhënave, të tilla si $ për stringat ose% për numrat.

Komentet

Dokumentacioni Software ekziston në dy forma, të jashtëm dhe të brendshëm. Dokumentacioni i jashtëm është ruajtur jashtë kodit burim, të tilla si specifikimet, fotografi ndihmë, si dhe dokumentet e projektimit. Dokumentacioni i brendshëm është i përbërë nga komentet që zhvilluesit shkruani në kodin burim në kohën e zhvillimit.

Një nga sfidat e dokumentacionit software është siguruar se komentet janë të mirëmbajtura dhe të përditësuar paralelisht me kodin burimor. Edhe pse duhet komentuar kodin burim i shërben asnjë qëllimi në kohë të drejtuar, ai është i paçmuar për një zhvillues i cili duhet të mbajë një pjesë veçanërisht të ndërlikuar ose të rëndë të softuerit.

Në vijim janë rekomanduar teknika komentimi:

Kur modifikoni kodin, gjithmonë mbani koment të përditësuar.

Në fillim të çdo rutine pune, ajo është e dobishme për të siguruar standarde, komente të njëllojtë, duke treguar qëllimin e punës, supozimet dhe kufizimet. Një koment unik duhet të jetë një hyrje të shkurtër për të kuptuar pse ekziston kjo punë apo ndërhyrje dhe çfarë mund të bëjë.

Shmangni duke shtuar komente në fund të një linj kodi; Komentet e end-line bëjnë kodin më të vështirë për t’u lexuar. Megjithatë, komentet fund të linjës janë të përshtatshme kur shfaqim deklarata ndryshueshme. Në këtë rast, lidh të gjitha komentet e end-line në një pozicion të përbashkët.

Shmangni përdorimin komenteve rrëmujë, të tilla si një linjë të tërë plot me ylla. Në vend të kësaj, përdorni hapësirën e bardhë për të ndarë komentet nga kodi.

Shmangni rrethimin e një blloku komenti me një kornizë tipografike. Mund të duket tërheqëse, por është e vështirë për të ruajtur.

Para vendosjes, duhet të kemi hequr të gjitha komentet e përkohshme ose pa lidhje për të shmangur konfuzionin përgjatë punës së ardhshme të mirëmbajtjes.

Nëse keni nevojë për komente për të shpjeguar një pjesë komplekse të kodit, shqyrtojeni kodin për të përcaktuar nëse ju duhet për rishkruar atë.Nëse është e mundshme, mos dokumentoni kod të keq, rishkruajeni atë. Edhe pse performanca nuk duhet të sakrifikohet zakonisht për të bërë kodin lehtë për syrin njerëzor, një ekuilibër duhet të mbahet në mes të performancës dhe mirëmbajtjes.

Përdorni fjali të plota kur shkruani komente. Komentet duhet të qartësojë kodin, nuk shtoni dykuptimësi.

Komentoni gjatë kodimit, për shkak se më shumë gjasa nuk do të ketë kohë për të bërë atë më vonë. Gjithashtu, duhet që ju të merrni një shans për të rihapur kodin që ju keni shkruar, atë që është e qartë sot ndoshta nuk do të jetë e qartë gjashtë javë nga tani.

Shmangni përdorimin e komenteve të tepërta ose të papërshtatshme, si vërejtjet humoristike.

Përdorni komente për të shpjeguar qëllimin e kodit. Ata nuk duhet të shërbejë si përkthimet fjalë për fjalë të kodit.

Koment asgjë që nuk është lehtësisht e dukshme në kodin.

Për të parandaluar përsëritjen e problemeve, gjithmonë përdorni komente në problemet e rregulluara dhe punimet rreth kodit, veçanërisht në një mjedis ekipi.

Përdorni komente mbi kodin që përbëhet nga pozicione dhe degë logjike. Këto janë fushat kryesore që do të ndihmojnë lexuesin kur lexoni kodin burim.

Ndani komentet me shenjat e hapjes (apo mbylljes) se komenteve me hapësirë ​​të bardhë. Duke bërë kështu do të bëhet që komentet të dalin të lehtë për të gjetur kur shihet pa patur ngjyra.

Gjatë aplikacionit, të ndërtohen komente duke përdorur një stil uniformë, me shenjat e pikësimit të qëndrueshme dhe strukturën.

Formati

Formatimi bën organizimi logjik i kodit të dal. Duke marrë kohë për të siguruar që kodi burim është i formatuar në mënyrë konsistente, logjike është e dobishme për veten tuaj dhe për zhvilluesit e tjerë të cilët duhet të deshifroj kodin burim.

Në vijim janë rekomanduar teknika formatimit:

Krijimi i një madhësi standarde për një nënpikën, të tilla si katër hapësira, dhe e përdorin atë në mënyrë të vazhdueshme. Vendose në pjesët e kodit duke përdorur vulë përshkruar.

Përdorni një font Monospace kur publikimit versionet hard-kopje e kodit burim.

Me përjashtim të konstanteve, të cilat janë të shprehura më të mirë në të gjitha karakteret uppercase me underscore, përdorin rastin e përzier në vend të underscore për të bërë emrat më të lehtë për të lexuar.

Vendose në formatimin e teksteve të hapura dhe të afërta vertikalisht ku palë tiranta përafruar, të tilla si:

for (i = 0; i <100; i ++)

{

...

}

Ju mund të përdorni një stil pjerrët, ku formatimin e teksteve hapur paraqiten në fund të rreshtit dhe formatimin e teksteve të ngushtë të shfaqet në fillim të linjës, të tilla si:

for (i = 0; i <100; i ++) {

...

}

Cilado style është zgjedhur, përdorni atë stil të gjithë kodin burim.

Krijimi i një gjatësi maksimale vijë për komente dhe kodit për të shmangur që të lëviz dritaren e kodit burim dhe për të lejuar për prezantim të pastër hard-kopje.

Përdorin hapësira para dhe pas shumicën e operatorëve kur kjo nuk ndryshon qëllimin e kodit. Për shembull, një përjashtim është simbol tregues përdoret në C ++.

Vendos një hapësirë ​​pas çdo presje në listat presje-kufizuar, të tilla si vlerat array dhe argumente, kur kjo nuk ndryshon qëllimin e kodit. Për shembull, një përjashtim është një argument Connection ActiveX® Data Object (ADO).

Përdorni hapësirë ​​të bardhë për të siguruar të dhëna organizative në kodin burimor. Duke bërë kështu krijon "paragrafët" të kodit, të cilat të ndihmuar lexuesit në të kuptuarit e Segmentimi logjik të softuerit.

Kur një vijë është e thyer në disa linja, e bëjnë të qartë se linja është i paplotë pa vijën e mëposhtme.

Kur është e përshtatshme, për të shmangur vendosjen e më shumë se një deklaratë për rresht. Një përjashtim është një lak në C, C ++, Visual J ++ ®, ose JScript®, të tilla si for (i = 0; i < 100; i++).

Kur shkruani HTML, të krijojë një format standard për tags dhe atributet, të tilla si duke përdorur të gjitha shkronja kapitale për klikoni dhe të gjitha Fjala për atributet. Si një alternativë, përmbahet specifikimeve XHTML për të siguruar të gjitha dokumentet e HTML janë të vlefshme.Edhe pse nuk janë madhësia e skedës tregtisë të humbura për të marrë parasysh kur krijimin e faqeve Web, përdorin vlerat e cituara atribut dhe tags përfundimtare për të lehtësuar maintainability.

Kur shkruani deklaratat SQL, përdorin të gjitha shkronja kapitale për fjalë kyçe dhe rast të përzier për elementet e bazës së të dhënave, të tilla si tavolina, kolona, ​​dhe pikëpamjet.

Ndani kodin burim logjike mes dosjeve fizike.

Në ASP, përdorni përndarësit script rreth blloqe të shkrimit në vend që rreth çdo linjë të shkrimit ose interspersing fragmente të vogla HTML me scripting server-side. Duke përdorur përndarësit script rreth çdo rresht, ose interspersing fragmente HTML me scripting server-side rrit frekuencën e kontekst kalimi në anën e serverit, e cila pengon punën dhe degradon kodin lexueshmëri.

Vënë çdo dispozitë të madh SQL në një linjë të veçantë kështu që deklaratat janë më të lehtë për të lexuar dhe të redaktoni, për shembull:

SELECT Firstname, Mbiemri

FROM Klientët

KU State = "WA"

Mos përdorni numra të mirëfillta ose strings mirëfillta, të tilla si For i = 1 To 7. Në vend të kësaj, përdorimi i quajtur konstante, të tilla si For i = 1 To NUM\_DAYS\_IN\_WEEK, për lehtësinë e mirëmbajtjes dhe të kuptuarit.

Pushim të mëdha, pjesët komplekse të kodit në të vogla, module kuptueshme.

Praktikat programim

Zhvilluesit përvojë të ndjekë praktika të shumta e programimit ose rregullat e gishtit, e cila zakonisht rrjedh nga mësimet e vështira-mësuar. Praktikat e listuara më poshtë nuk janë të gjithë-përfshirëse, dhe nuk duhet të përdoret, pa marrë në konsideratë. programuesit Veteran devijojnë nga këto praktika me rastin, por jo pa shqyrtim të kujdesshëm të pasojave të mundshme. Duke përdorur praktikën më të mirë programimit në kontekstin e gabuar mund të shkaktojë më shumë dëm sesa dobi.

Për të ruajtur burimet, të jetë selektiv në zgjedhjen e llojit të të dhënave për të siguruar madhësinë e një variable nuk është tepër i madh.

Mbani jetëgjatësinë e variablave si të shkurtër të jetë e mundur, kur variablat përfaqësojnë një burim i caktuar, për të cilat mund të ketë grindje, të tilla si një lidhje të bazës së të dhënave.

Mbani qëllimin e variablave si të vogla të jetë e mundur për të shmangur konfuzionin dhe për të siguruar maintainability. Gjithashtu, kur duke ruajtur kodin burimor trashëgimi, potenciali për pa dashje thyer pjesë të tjera të kodit mund të minimizohet, nëse objekti ndryshueshme është e kufizuar.

Përdorni variablave dhe rutinat për një dhe vetëm një qëllim. Përveç kësaj, për të shmangur krijimin e rutinat shumë qëllime që kryejnë një sërë funksionesh që nuk kanë lidhje.

Kur shkruani klasa, për të shmangur përdorimin e variablave publike. Në vend të kësaj, të përdorë procedurat për të siguruar një shtresë e encapsulation dhe gjithashtu për të lejuar një mundësi për të vërtetuar ndryshimet e vlerës.

Kur përdorni objekte të grumbulluara nga MTS, të marrë burimet e sa më vonë të jetë e mundur dhe për lirimin e tyre sa më shpejt të jetë e mundur. Si e tillë, ju duhet të krijojë objekte sa më vonë të jetë e mundur, dhe shkatërrimin e tyre sa më shpejt të jetë e mundur për të burimeve të lirë.

Kur duke përdorur objekte të cilat nuk janë duke u grumbulluara nga MTS, është e nevojshme për të shqyrtuar shpenzimet e krijimit të objektit dhe nivelin e grindjes për burime për të përcaktuar kur burimet duhet të jenë të fituar dhe u lirua.

Përdorni vetëm një skemë të transaksionit, të tilla si MTS apo SQL Server ™, dhe minimizuar qëllimin dhe kohëzgjatjen e transaksioneve.

Të jetë i kujdesshëm i përdorimit të variablave ASP Session në një mjedis të fermës Web. Në një minimum, mos vendosni objekte në variablat ASP Sesioni sepse shteti sesion është ruajtur në një makinë të vetme. Konsiderojnë ruajtjen shtet seancë në një bazë të dhënash në vend.

Komponentët pa shtetësi janë të preferuar, kur scalability apo të performancës janë të rëndësishme. Hartimi komponentët për të pranuar të gjitha vlerat e nevojshme si parametrave të dhëna në vend të duke u mbështetur mbi pronat objekt kur e quajtur metoda. Duke bërë kështu eliminon nevojën për të ruajtur objekt gjendjen mes thirrjeve metodë. Kur është e nevojshme për të ruajtur shtetin, e konsiderojnë duke përdorur metodat alternative, të tilla si ruajtjen e shtetit në një bazë të dhënash.

Mos të hapur lidhjet e të dhënave duke përdorur kredencialet e përdoruesit specifik. Lidhjet që janë hapur duke përdorur kredenciale të tilla nuk mund të grumbulluara dhe të ripërdoren, duke humbur kështu të mirat e bashkimit lidhje.

Shmangni përdorimin e detyruar të dhënat e konvertimit, nganjëherë referuar si detyrim të ndryshueshme apo hedh, të cilat mund të japin rezultate të papritura. Kjo ndodh kur dy ose më shumë variabla të llojeve të ndryshme të të dhënave janë të përfshirë në të njëjtën shprehje.Kur është e nevojshme për të kryer një cast për të tjera se një arsye të parëndësishme, se arsyeja duhet të jepet në një koment shoqërues.

Zhvillojnë dhe të përdorin rutinat gabim-trajtimin. Për më shumë informacion mbi trajtimin e gabimeve në Visual Basic, shih "Trajtimi Error dhe Debug" kapitull i / Visual Guide Microsoft Office 2000 Themelore programues, në dispozicion në Bibliotekën MSDN. Për më shumë informacion mbi trajtimin gabim dhe COM, shih ["Gabim Trajtimi"](https://msdn.microsoft.com/en-us/library/ms679692(v=vs.60).aspx) në Platforma SDK. Për më shumë informacion mbi gabim trajtimin për faqet Web, shihni [http://msdn.microsoft.com/workshop/author/script/weberrors.asp](https://msdn.microsoft.com/en-us/library/ms976140.aspx) .

Jini specifik, kur shpallja e objekteve, të tilla si ADODB.Recordsetnë vend të vetëm Recordset, për të shmangur rrezikun e perplasjeve emrin.

Kërkojnë përdorimin Option Explicitnë Visual Basic dhe VBScript për të inkurajuar maturi në përdorimin e variablave dhe për të minimizuar gabimet që rezultojnë nga gabimet tipografike.

Shmangni përdorimin e variablave me fusha e zbatimit.

Përdorni RETURNdeklaratat në procedurat e ruajtura për të ndihmuar programin e quajtur di nëse procedura ka punuar si duhet.

Përdorni teknika fillim detyruese kur është e mundur.

Përdorni Select Caseose Switchdeklarata në vend të kontrollit të përsëritura të një variable të përbashkët duke përdorur If…Thendeklarata.

Në mënyrë të qartë lirimin referenca objekt.