სამომხმარებლო ინტერფეისის ფუნქციონალური უზრუნველყოფა

გია მაქანდარაშვილი
Email: maqandarashvili@gmail.com

სამომხმარებლო ინტერფეისი ინტერფეისის ნაირსახეობაა, რომელშიც ერთი მხარე წარმოდგენილია ადამიანით (როგორც მომხმარებელი), მეორე მხარე კი მანქანა/მოწყობილობით, რომელიც თავის მხრივ წარმოადგენს საშუალებებისა და მეთოდების ერთობლიობას, რომელთა დახმარებითაც მომხმარებელი ურთიერთკავშირშია სხვადასხვა (ხშირად უფრო რთულ) მანქანებთან, მოწყობილობებთან და აპარატურასთან. ყველაზე ხშირად ეს ტერმინი კომპიუტერულ პროგრამებთან მიმართებაში იხმარება. ამ ტერმინში იგულისხმება საშუალებებისა და მეთოდების კრებული, რომელსაც ადამიანი მართავს როცა სისტემასთან კონტაქტშია. ინტერფეისი, რომელიც განსაზღვრავს მომხარებლის ინფორმაციულ რესურსებთან დამოკიდებულებას ქსელში ან კომპიუტერში. იგი კომპიუტერის (და არა მარტო კომპიუტერის) მომხმარებელს საშუალებას აძლევს ეურთიერთოს მას ბუნებრივი გზით. მაგალითად ხელის შეხებით, ხმით, ჟესტებით, მიმიკით, ისე, როგორც ურთიერთობს ადამიანი გარემო სამყაროსთან.

– – – – – – – – – – ◆ – – – – – – – – – –

სამომხმარებლო ინტერფეისი, მომხმარებლის სისტემასთან ურთიერთობის მნიშვნელოვანი საშუალებაა. აუცილებელია, განვასხვავოთ პროფესიონალი მომხმარებელი (ადამიანი, რომლიც კომპიუტერზე თავის პროფესიულ საქმიანობას ახორციელებს) და მომხმარებელი, რომლისთვისაც კომპიუტერთან ურთიერთობა ასე ვთქვათ, ნებაყოფლითია. პირველ კატეგორიას გააჩნია ინტერფეისთან შეგუების საშუალება, დრო და მოტივაცია, ამიტომ იგი ნაკლებად მგრძნობიარეა მსგავსი საკიტხების მიმართ. რაც შეხება მეორე კატეგორიის მომხმარებელს, მას აქვს მეტი არჩევანის უფლება და თუ ინტერფეისი მოხერხებული არაა, იგი არ ხარჯავს დროს და ძალისხმევას მასთან ურთიერთობის მექანიზმების დასაუფლებად. ნებისმიერი კარგი ინტერფეისი მომხმარებელს ფსიქოლოგიურ კომფორტის გრძნობას უქმნის. სამომხმარებლო ინტერფეისის შექმნის დროს უნდა: დამუშავდეს ინტერფეისის ელემენტების სისტემა, ურთიერთქმედების თავისებური კონცეფცია, რომლის შესწავლით მომხმარებელს შეეძლება იოლად აკეთოს ის, რაც მას სჭირდება; გამოინახოს როგორც ცალკეული ელემენტების, ისე მათი ჯგუფების გამოსახვის მორგებული ფორმა; აირჩეს გამოსახვის სტილი, რომელიც იოლი მისახვედრი და ესთეტიურად აღქმადი იქნება; მომხმარებელი სისტემასთან სამომხმარებლო ურთიერთობას ინტერფეისის გარემოსთან კავშირით იწყებს. ჩვენი აღქმა მოტივაციაზეა დამოკიდებული (გიბსონი. "ეკოლოგიური მიდგომა აღქმის ფსიქოლოგიისადმი"), ამიტომ სამომხმარებლო ინტერფეისის კონცეპტუალური დიზაინი გამოსაყენებელი პროგრამული უზრუნველყოფის იდეასა და კონცეფციას უნდა ემყარებოდეს. ე.ი. მომხმარებელი აღიქვამს მოლოდინებით განსაზღვრულ სიგნალებს და მათზე დაყრდნობით სხვადასხვა მოქმედებებს ახორციელებს. ასევე მნიშვნელოვანია ბალანსი იყოს დაცული: პროგრამის ინტერაქტიული შესაძლებლობებსა და მისი გამომსახველობითი საშუალებების სირთულეს შორის; პროგრამის ფუნქციურ, მანიპულირების შესაძლებლობებსა და მხატვრულად წარმოჩენას შორის; სამომხმარებლო ინტერფეისის ეფექტიანი გამოყენების საკითხების აქტუალობა იზრდება, ამიტომ მასთან დაკავშირებული საკითხები მნიშვნელოვანია. სამომხმარებლო ინტერფეისი თავისთავად გულისხმობს დიზაინის ხარისხის მახასიათებლებს. სამომხმარებლო ინტერფეისი და სარგებლიანობა ერთნაირად მნიშვნელოვანია მომხმარებლისთვის. პროგრამა, რომლიც რეალიზებას უკეთებს მოთხოვნებს, მაგრამ მასთან მუშაობა ინტერფეისის გამო რთულია თავისთავად არ არის ეფექტური და იგივე შეგვიძლია განვავრცოთ ერთ-ერთი საკვანძო საკითხზე-სარგებლიანობაზე, რომელიც დიზაინის ფუნციონალურ თვისებებს გულისხმობს. პროგრამის გამოყენების ეფექტიანობა აუცილებლად უნდა ითვალისწინებდეს პასუხს კითხვაზე: რამდენად შეესაბამება ის მომხმარებლის მოთხოვნებს? ასრულებს თუ არა მომხმარებლისთვის საჭირო ოპერაციებს.

სამომხმარებლო ინტერფეისი, გულისხმობს როგორც პროგრამული უზრუნველყოფის გამოყენების ეფექტიანობას, მოხერხებულობას, სარგებლიანობას, ასევე, მისი დამუშავების პროცესში გამოყენებულ მეთოდებს. საჭირო რესურსებთან მუშაობის დროს იკვეთება პრობლემები, რომელიც ეხება პროგრამული უზრუნველყოფის გამოყენების მოხერხებულობას, კომფორტულობას (მაგ: საიტის გვერდის ჩატვირთვის პერიოდი, ნავიგაციის მოხერხებულ საშუალებები, რესურსის ოპტიმალურ სტრუქტურა, გაფორმების, ინტერფეისის ეფექტურობა და ა.შ. რომლის მეშვეობით მომხმარებელი პოულობს მისთვის საჭირო ინფორმაციას). მომხმარებლის მიერ ინტერფეისის გამოყენების ძირითადი პრობლემა მომხმარებლის ყურადღებისა და სისტემის ატიურობის წერტილების

სინქრონიზებაა. გამოვყოთ სამომხმარებლო ინტერფეისის ეფექტურობის შემდეგ თვისებები: მარტივი ნავიგაცია, ეფექტურობა, აღქმა, მომხმარებლის მიერ დაშვებული შეცდომების გავლენა ურთიერთობის პროცესზე, მოთხოვნის დაკმაყოფილება.

მოვიყვანოთ მაგალითი: თუ ვებ-თან მუშაობა რთულია მისი ინტერფეისის გამო ან იგი ვერ პასუხობს კლიენტის მოთხოვნებს მომხმარებელი ტოვებს გვერდს. თითქმის არ არსებობს მომხმარებელი, გამონაკლისის შემთხვევაში, რომელიც დიდ დროს ხარჯავს კონკრეტული ინტერფეისის შესასწავლად . პირველივე სირთულის შემთხვევაშიც ხდება საიტის დატოვება. ამიტომ უნდა გავითვალისწინოთ, თუ გვინდა ჩვენმა ინფორმაციამ /პროდუქტმა მიაღწიოს მომხმარებლამდე, მან უნდა იპოვოს ის. არსებობს სამომხმარებლო ინტერფეისის ანალიზის/ტესტირების სხვადასხვა გზები. განვიხილოთ რამოდენიმე მათგანი: პოტენციური კლიენტის მიერ, ზემოდ ჩამოთვლილი საკითხების შეფასება/ტესტირება; მომხმარებლის მიერ საიტის ფარგლებში შესრულებული სამუშაოს ანალიზი , საიტზე მომხმარებლის შემოსვლისა და დაყოვნების დროის ანალიზი; მიზნობრივი აუდიტორიის მოლოდინებისა და მოთხოვნების რეალიზების ანალიზი; ამგვარი მარკეტინგული აუდიტის შედეგად მიღებული ინფორმაცია გამოიყენება საიტის აუდიტორიის შესწავლის, სრულფასოვანი მახასიათებლების განსაზღვრისა და ინტერნეტ-პროდუქტის ეფექტურობის გაზრდისათვის. ტესტირებით ჩვენ ვღებულობთ, არა მარტო რეკომენდაციებს, არამედ ვითვალისწინებთ მათ რეკონსტრუქციის დროს. სამომხმარებლო ინტერფეისის ფუნდამენტად მოიაზრება დისციპლინა HCI (human - computer interaction - ადამიანისა და კომპიუტერის ურთიერთქმედება), რომლის შესწავლის მიზანია კომპიუტერული სისტემებისა და მათი გამოყენების საკითხები, რათა უზრუნველყოფილი იქნას ინტერაქტიული ურთიერთობის ეფექტურობა.

ვებ-პროგრამირება არის სერვერული პროგრამული გადაწყვეტა, რომელიც შესაძლოა განთავსდეს ინტერნეტ სივრცეში ან დახურულ ქსელში. ვებ - პროგრამირებით შესაძლებლია როგორც მარტივი ვებ - ამოცანების გადაწყვეტა ასევე ნებისმიერი სახის ბიზნეს ამოცანის, სოციალური სფეროს, ფინანსური ამოცანები ან/და სხვა სფეროს პროცესების ავტომატიზაცია. ვებ - პროგრამირება აერთიანებს რამდენიმე ნაწილს, ესენია: კლიენტის მხარის პროგრამირება, ვებ-დიზაინი, ვებ - კონტენტების მართვა, სერვერის მხარეს პროგრამირება, კლიენტ-სერვერ შორის კავშირი, ქსელური პროგრამირება, სერვისზე ორიენტირებული პროგრამირება, უსაფრთხოება და სხვა;
კლიენტი წარმოადგენს იმ მომხმარებელს, რომელიც უკავშირდება სერვერს გარკვეული მომსახურების მისაღებად. სერვერული მომსახურების მისაღებად საჭიროა შესაბამისი პროგრამული უზრუნველყოფა. დღეისთვის, ვებ - პროგრამირებისთვის გამოიყენება ინტერნეტ მიმომხილველები, რომელთა როლია დაუკავშირდენა სერვერს და მიიღონ შესაბამისი ინფორმაცია. იმისათვის, რომ მათ მოახერხონ ამ მიღებული ინფორმაციის ვიზუალიზაცია, ამისთვის გამოიყენებენ HTML ფორმატს, რომელიც გარდაქმნის სტრუქტურულ ტექსტურ ინფორმაციას გარკვეულ ვიზუალად, ფორმად, სიად, სურათათ და აშ. HTML - ის განვითარებას ასევე ერთვის CSS ვიზუალური მხარის დამუშავებისთვის და სკრიპტული (Script) პროგრამული ენა კლიენტის მხარეს ფუნქციონალური თვისებების გასამდიდრებლად; ასევე შესაძლებელია გამოყენებული იყოს სხვადასხვა ტიპის ფორმატები, რომელთა ვიზუალიზაციას დამოკიდებულია კლიენტის პროგრამულ ნაწილზე.
ვებ პროგრამირებაში სამომხმარებლო ინტერფეისის ფუნქციონალის უზრუნველსაყოფად გამოიყენება სხვადასხვა ჰიპერტექსტური მარკირების და პროგრამირების ენები განვიხილოთ რამოდენიმე მათგანი
HTML - ინტერნეტში არსებული ნებისმიერი ინფორმაცია, რომლთა დათვალიერება შესაძლებელია ინტერნეტ-ბრაუზერების საშუალებით, ინახება რომელიმე ვებ სერვერზე საიტის სახით. ვებ საიტი შედგება ვებგვერდებისაგან, რომელთა მარკირება და ერთმანეთთან დაკავშირება ხდება HTML (Hyper Text Markup Language) ჰიპერტექსტების მარკირების ენის საშუალებით. HTML ინტერნეტის ფუნდამენტურ საბაზო ტექნოლოგიას წარმოადგენს. HTML ენა ბრიტანელმა ფიზიკოსმა ტიმ ბერნს ლიმ (Tim Berners-Lee) შეიმუშავა 1989 წელს ჟენევაში. თავდაპირველად ის მეცნიერულ-ტექნიკური ინფორმაციის გასაცვლელად შეიქმნა, მას შემდეგ მუდმივად ვითარდება. ვებგვერდის ბრაუზერში ასახვისათვის გამოიყენება HTML ენა, რომლის საშუალებით ხორციელდება ვებ-გვერდის სტრუქტურის (Layout) შექმნა, ხოლო გაფორმება ანუ ვიზუალური სახის მიცემა ხდება CSS (Cascading Style Sheets - კასკადური სტილების ცხრილები) სტილების საშუალებით. HTML დაწერილია HTML ელემენტების ფორმით, რომელიც შედგება ტეგებისგან, რომლებიც გვერდის შინაარსში მოქცეულია კუთხიან ფრჩხილებში (როგორც <html>). HTML ტეგები ძირითად წყვილად არის წარმოდგენილი, როგორც <h1> და </h1>, თუმცა ზოგიერთი ტეგი ცარიელ ელემენტს წარმოადგენს და ამიტომ არაწყვილია, მაგალითად <img>. წყვილის პირველი ტეგი არის საწყისი ტეგი, ხოლო მეორე - დაბოლოება. მათ შორის დიზაინერები ამატებენ ტექსტს, სხვა ტეგებს, კომენტარებს და ტექსტზე დაფუძნებული შინაარსის სხვა ტიპებს. ბრაუზერის დანიშნულებაა HTML დოკუმენტების წაკითხვა და მათი ხილვადი ან მოსმენადი გვერდების სახით გამოტანა. ბრაუზერი არ აჩვენებს HTML ტეგებს, იგი მათ იყენებს გვერდის შიგთავსის ასახვისთვის. HTML ელემენტები წარმოადგენენ ყველა საიტის შემადგენელ ნაწილს. HTML-ის მეშვეობით დასაშვებია სურათებისა და ობიექტების გვერდზე ასახვა, ისევე, როგორც ინტერაქტიული ფორმების შექმნა. შესაძლებელია სტრუქტურული დოკუმენტების შექმნა, რაც ხდება ტექსტის სტრუქტურული სემანტიკის აღნიშვნით, მაგალითად აბზაცებით, სიებით, სათაურებით, ბმულებით, ციტირებებით და სხვა ელემენტებით. HTML- ში შესაძლებელია სხვადასხვა სკრიპტის თანდართვაც, რომლებიც დაწერილია ჯავასკრიპტის მსგავს ენებზე. ამით ხდება HTML გვერდზე ასახული მოქმედებების კონტროლი.

CSS – ვინაიდან HTML - ის ენა მოკლებულია იმ შესაძლებლობებს, რომ გავაფორმოთ დოკუმენტის ვიზუალური მხარე ისე როგორც ეს თანამედროვეობას შეესაბამება, ამისათვის WEB-გვერდების ვიზუალიზაციაში აუცილებლობა წარმოიშვა შეექმნათ დამატებითი ენა რომელსაც ეწოდა CSS(Cascading Style Sheet) ანუ კასკადური სტილების ენა. CSS კასკადური სტილების ენა საშუალებას გვაძლევს დოკუმენტის ვიზუალიზაცია მომხმარებლისთვის იყოს კომფორტული, ადვილად აღსაქმელი და რაც ყველაზე მთავარია დოკუმენტი ყველა მოწყობილობაზე აისახებოდეს ჩვენთვის სასურველი ფორმით. CSS ენის გამოჩენამ WEB ტექნოლოგიების სივრცეში, მისცა დეველოპერებს საშუალება შეექმნათ უფრო მიმზიდველი და ეფექტური საიტები. აღნიშნული ენის დანერგვამ მკვეთრად გამიჯნა ერთმანეთისგან HTML დოკუმენტის სტრუქტურული ნაწილი მისი ვიზუალური ნაწილისგან. როდესაც ამ ენის შექმნამდე ვიყენებდით ათასგვარ ხრიკს, რომ დოკუმენტი ვიზუალურად დაგვეფორმატებინა ჩვენი სურვილის და მიხედვით, ახლანდელ დროში ეს პრობლემა აღმოიფხვრა CSS ენის წყალობით. ანუ კიდევ ერთხელ აღვნიშნოთ, რომ HTML კოდს ვიყენებთ რეალურად დოკუმენტის სტრუქტურის შექმნაში და CSS კოდს ვიყენებთ ცალსახად მხოლოდ მის ვიზუალურ ფორმირებაში და მიმზიდველი ეფექტების შექმნისთვის. აქვე ერთ გარემოებას გავუსვათ ხაზი, CSS ენა არავითარ შემთხვევაში არ წარმოადგენს HTML ენის შემცვლელ ალტერნატიულ ტექნოლოგიას. როგორც ზემოთ აღვნიშნეთ ეს ორი კოდირების ენა ერთმანეთს მკვეთრად ემიჯნება თავისი დანიშნულების მიხედვით. ახლა მოდით აღვწეროთ თუ რის გამო არის CSS - ი HTML კოდირების ენისგან განსხვავებული დოკუმენტის ვიზუალიზაციის ნაწილში. როგორც წესი მოგეხსენებათ HTML ენაში დოკუმენტის სტრუქტურა იქმნება ტეგების საშუალებით და თითოეული დოკუმენტის ელემენტს, იქნება ეს გრაფიკული გამოსახულება თუ ტექსტური ელემენტი, გარკვეული ატრიბუტის (პარამეტრის) მეშვეობით ენიჭება დაშვებული მნიშვნელობები, როგორიცაა სიგანე, სიმაღლე, ტექსტის ფერი და ასე შემდეგ. CSS - ს შეიძლება გავიგოთ როგორც რევოლუციური ნაბიჯი ვებ-დიზაინის სამყაროში. რადგან მან გადაჭრა ყველა დიზაინის პრობლემა სხვადასხვა ბრაუზერებთან მიმართებაში, ამის გარდა მას გააჩნია უამრავი უპირატესობა, მაგალითად სტილის კონტროლი ყველა დოკუმენტზე მხოლოდ ერთი კასკადური სტილის ფურცლიდან, ერთი მითითებების ნაკრები შეგვიძლია გამოვიყენოთ კასკადურად ერთი ჯგუფის ქვეშ გაერთიანებული ტეგებისთვის, დიზაინის უფრო დახვეწილი მენეჯმენტი, უამრავი გართულებული და წინ წაწეული დიზაინის ტექნიკები და კიდევ მრავალი.

სტატიკური ვებ გვერდების ერა დიდი ხანია დასრულდა. დღეს მსოფლიოს დიდი ნაწილი ორიენტირებულია დინამიური საიტების, სხვადასხვა ეფექტებითა და ინტერაქციით გაფორმებული საიტების შექმნაზე.ვებსაიტზე ინტერაქტივისა და ეფექტების შექმნის საშუალებას იძლევა Javascript-ი.

Javascript - პროგრამირების ერთ-ერთი ფართოდ გავრცელებული ენაა ის მსოფლიოს ყურადღების ცენტრში 1995 წლიდან მოექცა. მისი მეშვეობით შესაძლებელი იყო ვებ გვერდზე პროგრამული ნაწილის განთავსება. საწყის ეტაპზე იგი აღიქმებოდა მხოლოდ ბრაუზერ Netscape Navigator - ში.ამ ენაზე მოთხოვნის სწრაფმა ზრდამ გამოიწვია უმოკლეს დროში მისი ინტეგრაცია წამყვან ბრაუზერებში. ამ პროგრამირების ენამ საშუალება მისცა დეველოპერებს შეექმნათ ვებ საიტებისთვის სხვადასხვა პროგრამული დანამატები (მაგ. რუკები), მაგრამ ძირითადად იგი გამოიყენებოდა ვებ გვერდზე ინტერაქციისა და სხვადასხვა ეფექტების შესამუშავებლად. Javascript ფართო შესაძლებლობების მქონე პროგრამირების ენაა, მისი შესაძლებობლები დამოკიდებულია გარემოზე რომელშიც უწევს ფუნქციონირება. მაგ. : ბრაუზერში მას შეუძლია ყველაფერი რაც შეეხება ვებ გვერდის მანიპულაციას, მომხმარებელს, ურთიერთობას სერვერთან გარკვეული დოზით:

- ახალი HTML ტეგების შექმნა, არსებულის წაშლა, ტეგებისათვის სტილების გაწერა, მათი დამალვა/გამოჩენა;
- მომხმარებლების მოქმედებებზე რეაგირება, მაუსისა და კლავიატურის ხდომილებებზე (event) კონრეტული ფუნქციონალის მინიჭება;
- სერვერზე მოთხოვნების გაგზავნა, ინფორმაციის ჩატვირთვა ვებ გვერდის განახლების გარეშე (ajax ტექნოლოგია);
- Cookie - თან ურთიერთობა;

ეს მხოლოდ ნაწილია იმ შესაძლებლობისა, რომელსაც ფლობს პროგრამირების ენა Javascript. არსებობს Javascript - ის უამრავი ბიბლიოთეკა, რომელიც ამარტივებს ბევრად მუშა პროცესს, რადგანაც ორიენტირებულნი არიან კონკრეტული მიმართულებების გასამარტივებლად. თანამედროვე ბიბლიოთეკების დახმარებით შესაძლებელია მხოლოდ javascript-ის საშუალებით შეიქმნას ვებ გვერდები.

ლიტერატურა

[1] Adham Dannaway, “Designing interfaces”
[2] Everett N McKay, “UI is Communication”
[3] Krug C, “Web-design”