2025

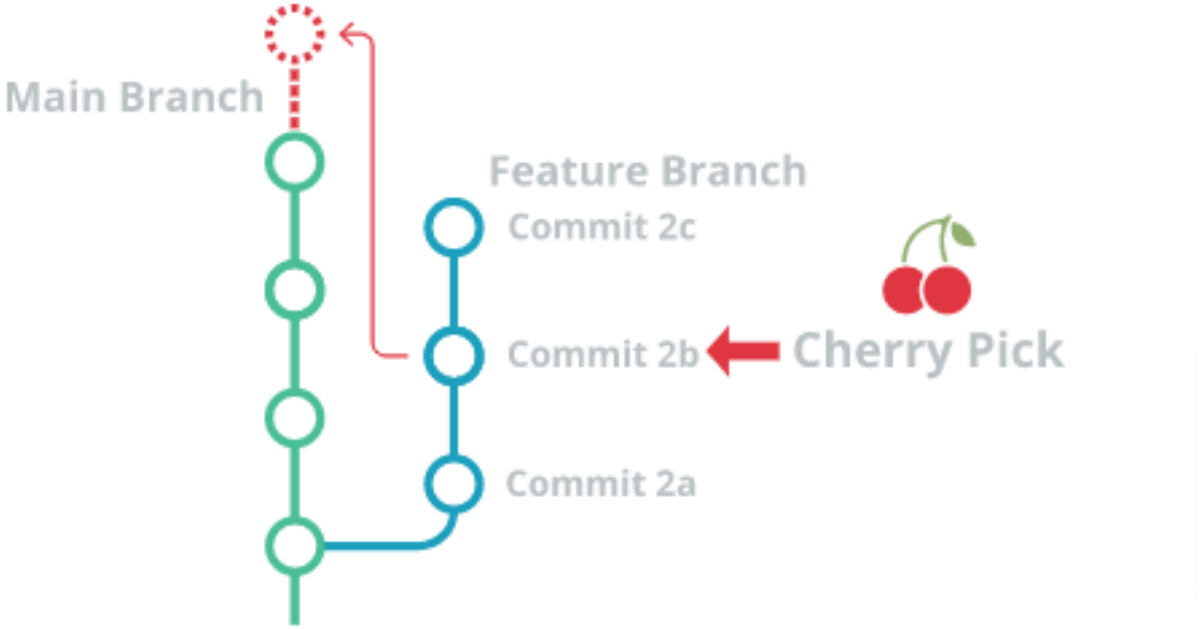
ბიზნესისა და ტექნოლოგიების უნივერსიტეტი

ვერსიონირება და უწყვეტი ინტეგრაცია

თემა მეექვსე და მეშვიდე:
ცვლილებების შერჩევა Cherry-pick-ის მეშვეობით, .gitignore-ის გამოყენება, Fork-ის დანიშნულება და Pull Request-ის პროცესი

Cherry-pick და ცვლილებების შერჩევა

git cherry-pick არის ბრძანება, რომელიც კონკრეტულ კომიტს ან კომიტების ჯგუფს აკოპირებს სხვა ბრენჩზე. ეს ბრძანება განსხვავდება merge-ისგან იმით, რომ ის არ აერთიანებს ორ მთლიან ბრენჩს, არამედ ირჩევს ინდივიდუალურ ცვლილებებს, რომლებიც კონკრეტულ კომიტშია ასახული. Cherry-pick-ის გამოყენება მიზანშეწონილია: თუ რაიმე ბაგ ფიქსის გადატანაა საჭირო ერთი ბრენჩიდან მეორეზე დამერჯვის გარეშე, თუ ბრენჩზე ვმუშაობთ პროგრამის ფუნქციონალზე რომელიც ბევრ საკითხს მოიცავს და საჭიროა კონკრეტული ფუნქციონალის გადატანა ან მთლიანი ფუნქციონალის ნაწილ ნაწილ გადატანა ძირითად შტოზე. ამ მიზეზების გარდა შესაძლებელია სხვა სპეციფიური გარემოება არსებობდეს რის გამოც დამერჯვას Cherry-pick-ის გამოყენება აჯობებს.



Cherry-pic-ის გამოსაყენებლად საჭიროა ვიმყოფებოდეთ იმ შტოზე სადაც გვინდა ქომითის დამატება და მივუთითოთ ქომითის ჰეში რომლის დამატებაც გვინდა. მაგალითად:

git cherry-pick <ქომითის ჰეში>

git cherry-pick <პირველი ქომითის ჰეში> <მეორე ქომითის ჰეში>

Cherry-pick-ის პროცესში შესაძლებელია კონფლიქტების წარმოშობა, განსაკუთრებით იმ შემთხვევაში, თუ გადმოტანილი ცვლილებები ეწინააღმდეგება მიმდინარე ბრენჩის

ინფორმაციას. კონფლიქტის შემოწმება შესაძლებელია **Status** ბრძანების მეშვეობით და მისი მოგვარებაც არაფრით განსხვავდება ჩვეულებრივი კონფლიქტისგან, თუმცა, კონფლიქტის მოგვარების შემდეგ საჭირო იქნება Cherry-pick ის გაჩერებული პროცესის გაგრძელება ან ცვლილების გაკეთება თუ არ იქნება მიზანშეწონილი მისი გაუქმება.
ამისთვის გამოიყენება შემდეგი ბრძანებები:

git cherry-pick --continue

git cherry-pick --abort

git cherry-pick წარმოადგენს ეფექტურ ინსტრუმენტს, რომელიც საშუალებას იძლევა სამუშაო პროცესში მოქნილად და მიზნობრივად გავაკონტროლოთ ცვლილებების გადაცემა. მისი სწორად გამოყენება ამცირებს merge-ით გამოწვეულ რისკებს.

.gitignore-ის გამოყენება Git-ში

.gitignore წარმოადგენს სპეციალურ ფაილს Git-ის ეკოსისტემაში, რომლის მიზანია განსაზღვროს, რომელი ფაილები ან დირექტორიები უნდა გამოირიცხოს ვერსიის კონტროლიდან. მისი გამოყენება საშუალებას იძლევა არამნიშვნელოვანი, დროებითი ან პირადი ფაილები არ მოხვდეს Git-ის რეპოზიტორიის ისტორიაში, რითაც შენარჩუნდება სისუფთავე, სიმარტივე და უსაფრთხოება პროექტში.
.gitignore არის უბრალო ტექსტური ფაილი, რომელიც მდებარეობს პროექტის მთავარ დირექტორიაში (ან ქვედირექტორიებში) და მასში ჩამოწერილია პატერნები ანუ წესები, რომელთა საფუძველზეც Git-ი იგნორირებას უკეთებს შესაბამის ფაილებს. პატერნების მაგალითია:

- **File.txt** - კონკრეტული ფაილის მითითება
- ***.txt** - ყველა .txt გაფართოების ფაილის მიტითება
- **/Pictures/** - “Pictures” საქაღალდის მითითება

ასევე არსებობს სხვა სიმბოლოებიც რომლებიც იშვიათად გამოიყენება, განვიხილოთ ყველა სიმბოლო:

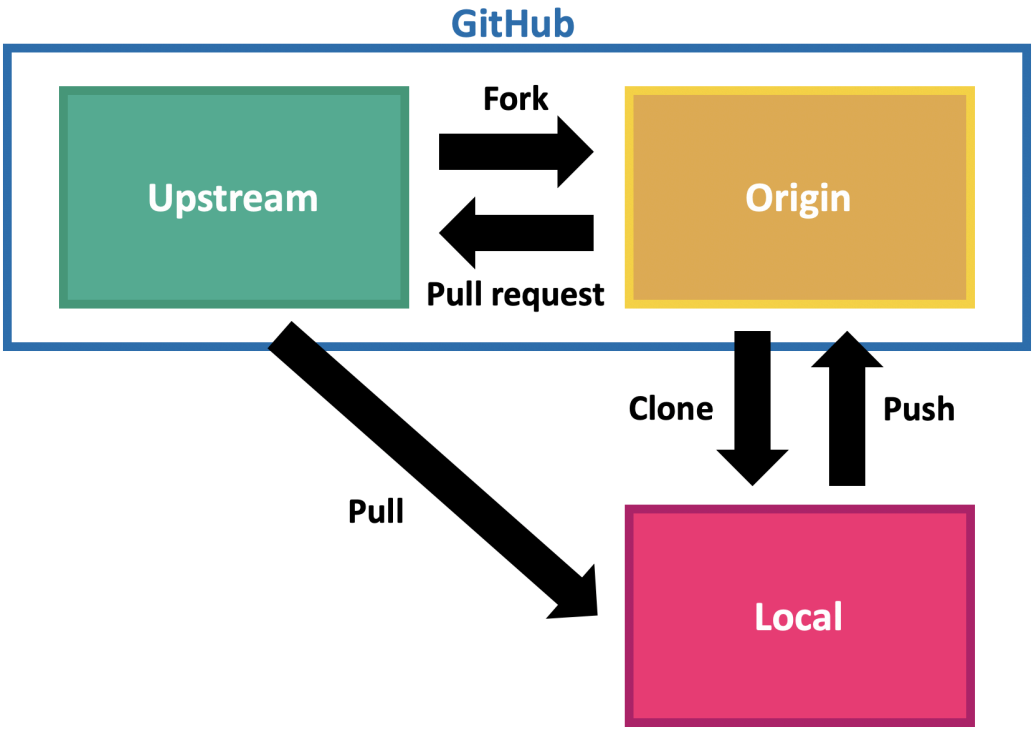
- * - 0 ან მეტი სიმბოლო
- ? - 1 სიმბოლო
- / - მითითება დირექტორიაზე
- # - კომენტარი
- ! - გამონაკლისი

.gitignore აუცილებელი ინსტრუმენტია და მისი გამოყენება უმჯობესია პირველივე ქომითის შექმნასთან. Gitignore ინდივიდუალურ მორგებას საჭიროებს, ძირითადად მასში იწერება პაკეტ მენეჯერების დირექტორიები, პირადი ან კონფიდენციალური ინფორმაცია, სისტემური ფაილები, სხვადსხვა რედაქტორების ფაილები და სხვა.

Fork-ის დანიშნულება და Pull Request-ის პროცესი

Fork არის რეპოზიტორიის პირადი ასლი. მასში ცვლილებების შენახვა ორიგინალური რეპოზიტორიისგან დამოუკიდებლად ხდება. ძირითადად მას იყენებენ დიდ პროექტებში როდესაც ძირითად შტოზე წვდომა შეზღუდულია ან Open Source პროექტებში.
Fork-თან მუშაობა ჩვეულებრივი რეპოზიტორიის მუშაობის იდენტურია. როდესაც fork-ზე მუშაობა დასრულდება, საჭიროა ცვლილებების გადატანა ორიგინალურ რეპოზიტორიაზე და ამისთვის pull request-ი გამოიყენება

Pull request ნიშნავს ინიციატივას, whereby fork-იდან ან branch-იდან ცვლილებები დაბრუნდეს ორიგინალურ რეპოზიტორიაში.



Pull request-ი გამოიყენება მაშინ როდესაც ორიგინალურ რეპოზიტორიას ცვლილებების ჩაწერაზე შეზღუდვა ადევს. მსგავსი დაცული სტრუქტურის დროს, pull request არის ერთგვარი შეთავაზება რომელსაც მომხმარებელი (ის ვინც Fork-ი გააკეთა) სთავაზობს მეპატრონეს (ორიგინალური რეპოზიტორიის მფლობელს). მეპატრონეს აქვს საშუალება უარყოს ან დაადასტუროს მოთხოვნა რის შემდეგაც ცვლილებები ორიგინალურ რეპოზიტორიაში ჩაიწერება. აღსანიშნავია რომ მსგავსი სტრუქტურისთვის ორიგინალი რეპოზიტორია აუცილებლად public უნდა იყოს.

Pull request-ს გააჩნია რამოდენიმე მნიშვნელოვანი ატრიბუტი, რომლის მითითებაც საჭიროა კორექტული რექვესთის გასაგზავნად. მაგალითად:

- **Title** - სათაური
- **Description** - დეტალური აღწერა
- **Reviewers/Assignees** - ვისაც ვუგზავნით

Fork და Pull Request აყალიბებს Git-ის (და Github-ის) თანამშრომლობის ყველაზე გავრცელებულ და უსაფრთხო მოდელს. ისინი ქმნიან დეცენტრალიზებული თანამშრომლობის სისტემას, რაც შესაძლებელს ხდის დიდი და მცირე პროექტების მართვას ღია გარემოში.