



**UNIVERZITET U NOVOM
SADU PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA MATEMATIKU I INFORMATIKU**

Iceberg i Baseline

- Seminarski rad iz predmeta Live Object Programming in Pharo -

Nikola Kampfer 43/19

Novi Sad, 2024

Sadržaj

1. Uvod.....	3
2. Baseline.....	4
2.1 Kreiranje Baseline-a.....	5
2.2 Definisanje paketa.....	6
3. Iceberg.....	7
3.1 Korišćenje Iceberg-a.....	7
3.2 Dodavanje paketa.....	9
3.3 Commit.....	10
4. Kreiranje repozitorijuma na GitHub-u.....	11
5. Baseline, Iceberg i GitHub.....	12
6. Zaključak.....	13
7. Literatura.....	14

1.Uvod

Pharo je dinamičan objektno-orijentisani programski jezik i razvojno okruženje zasnovano na programskom jeziku Smalltalk. Namenjen je razvoju softverskih rešenja koja se lako održavaju i proširuju. U ovom radu ćemo se fokusirati na dve komponente Phara: Iceberg i Baseline.

Pharo je poznat po svojoj interaktivnosti i fleksibilnosti, što ga čini idealnim za početnike i istraživače. Svaka linija koda može biti odmah evaluirana, što omogućava brzo testiranje i ispravljanje grešaka.

2.Baseline

Baseline je alat u Pharo-u koji služi za upravljanje zavisnostima u projektu. Omogućava programerima da definišu koje biblioteke i paketi su potrebni projektu, čime se olakšava instalacija i održavanje. Na primer, pomoću Baseline-a možeš da opišeš koji se delovi koda učitavaju i odakle, što omogućava laku reprodukciju okruženja na drugim računarima ili pri radu u timovima. Koristi se sa Metacello objektom za učitavanje projekata koji imaju definisan Baseline. Baseline omogućava lakše definisanje zavisnosti i reprodukciju projektnog okruženja na drugim računarima ili u timskom radu.

Pharo projekti često zahtevaju konfiguraciju kako bi se definisalo kako treba da budu učitani, a to se postiže putem Baseline-a. Baseline definiše pakete projekta, njihove međusobne zavisnosti, kao i zavisnosti prema spoljnim projektima i nezavisne podgrupe koje mogu da se učitaju.

Dodavanje Baseline-a u projekat ima nekoliko prednosti:

- Omogućava lakše učitavanje projekta.
- Olakšava drugima da doprinesu projektu.
- Korisnicima omogućava da ne moraju da znaju zavisnosti projekta.
- Jasno definiše zavisnosti projekta.
- Osigurava da se paketi i zavisnosti učitavaju ispravnim redosledom.

2.1 Kreiranje Baseline-a

Da bi se napravio Baseline, potrebno je kreirati paket nazvan *BaselineOf<ImeProjekta>* i odgovarajuću klasu koja nasleđuje *BaselineOf*. Na primer, za projekat Counter, kreira se paket i klasa nazvani *BaselineOfCounter*.

```
BaselineOf subclass: #BaselineOfCounter
  instanceVariableNames: ''
  classVariableNames: ''
  package: 'BaselineOfCounter'
```

Slika 1

Zatim kreiramo metod *BaselineOfCounter >> baseline:* sa sledecim sadrzajem:

```
baseline: spec
  <baseline>
  spec for: #common do: [
    "Packages"
    spec
      package: 'Counter';
      package: 'Counter-Tests' with: [ spec requires: #('Counter') ] ].
```

Slika 2

Ako je projekat uskladisten sa metapodacima, potrebno je dodati ovaj metod:

```
projectClass
  ^ MetacelloCypressBaselineProject
```

Slika 3

2.2 Definnisanje paketa

Da bi definisali paket u projektu , potrebno je poslati poruku `#package:` specifikaciji sa imenom tog paketa kao argument.

```
baseline: spec
  <baseline>
  spec
    for: #common
    do: [
      "Packages"
      spec
        package: 'MyProject';
        package: 'MyProject-Tests';
        package: 'MyProject-Gui';
        package: 'MyProject-Gui-Tests';
        package: 'MyProject-Examples' ]
```

Slika 4

Definisanje paketa nije dovoljno za njihovo učitavanje jer neki mogu zavisiti od drugih. Na primer, *MyProject-Tests* treba učitati nakon *MyProject*. Zavisnosti između paketa projekta mogu se upravljati pomoću poruke `#package:with:`, koja pruža specifične informacije o zavisnostima unutar projekta.

Za upravljanje eksternim zavisnostima, potrebno je koristiti dodatne konfiguracije. Ovaj pristup osigurava da se svi delovi projekta učitaju u ispravnom redosledu, što je ključno za stabilnost i pravilno funkcionisanje aplikacije.

Slika 5

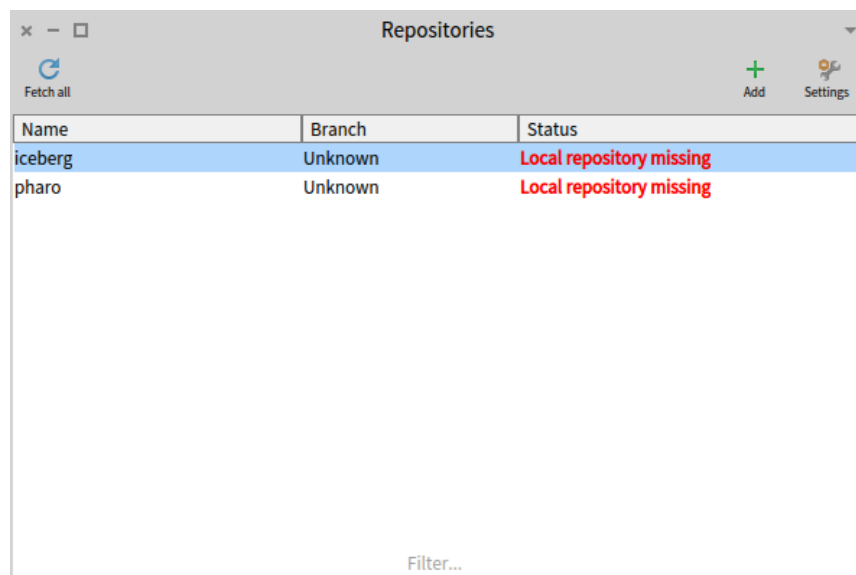
```
baseline: spec
  <baseline>
  spec
    for: #common
    do: [
      "Packages"
      spec
        package: 'MyProject';
        package: 'MyProject-Tests' with: [ spec requires: #('MyProject') ];
        package: 'MyProject-Gui' with: [ spec requires: #('MyProject') ];
        package: 'MyProject-Gui-Tests' with: [ spec requires: #('MyProject-Tests') ];
        package: 'MyProject-Examples' with: [ spec requires: #('MyProject-Gui') ] ]
```

3. Iceberg

Iceberg je alat za upravljanje verzionisanjem koda u Pharo-u, koji koristi Git kao sistem kontrole verzija. Omogućava programerima da direktno iz Pharo okruženja rade sa Git repozitorijumima – mogu klonirati projekte, praviti grane, commit-ovati promene i upravljati istorijom verzija. Iceberg povezuje Pharo sa platformama kao što su GitHub, GitLab i Bitbucket, olakšavajući saradnju i održavanje koda.

3.1 Korišćenje Iceberg-a

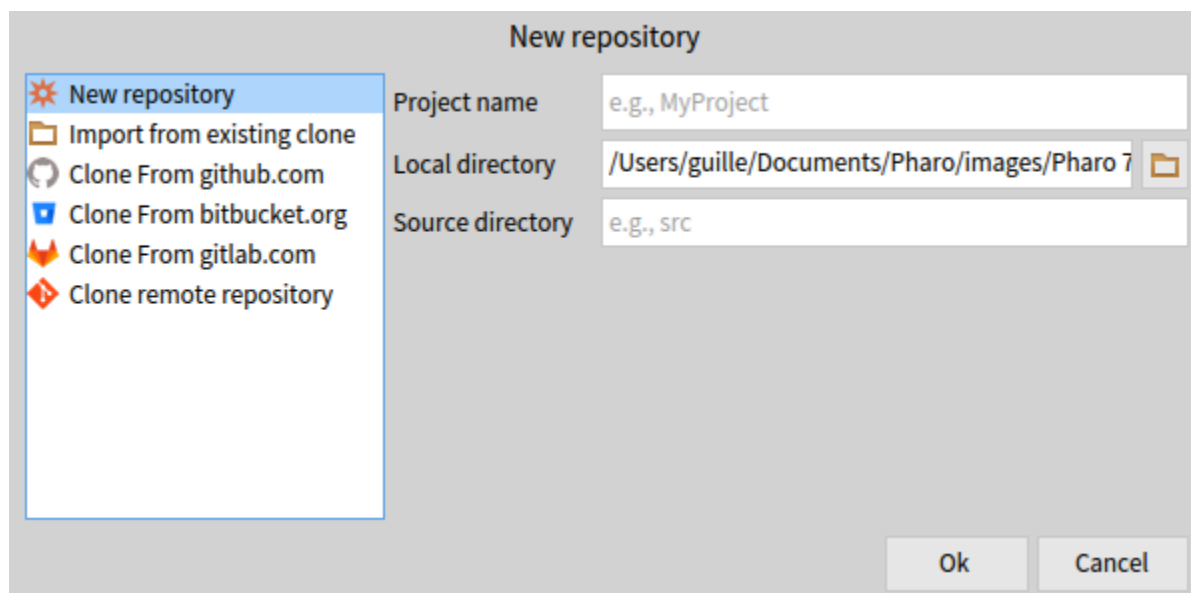
Pre nego što se krene sa Iceberg-om, potreban je Git repozitorijum. On se kreira na GitHub-u. Da bi se otvorio Iceberg, potrebno je otici na Pharo-ov meni, pod 'Tools', i izabrati 'Repositories Browser'. Ovo će omogućiti pristup alatima za upravljanje repozitorijumima direktno iz Pharo okruženja, olakšavajući rad sa Git-om.



Slika 6

Browser repozitorijuma u Iceberg-u prikazuje sve repozitorijume instalirane u vašem Pharo okruženju, bilo da su povezani sa Git repozitorijumima na vašem računaru ili ne. Podrazumevano, Iceberg dolazi sa nekim pre-konfigurisanim repozitorijumima koji omogućavaju doprinos Pharo-u i Iceberg-u. Da biste dodali novi repozitorijum, kliknite na dugme Add, a zatim unesite informacije o repozitorijumu koji želite da klonirate, uključujući izvor kloniranja.

Ovo omogućava lak pristup verzionisanju i doprinosu projektima direktno iz Pharo-a.

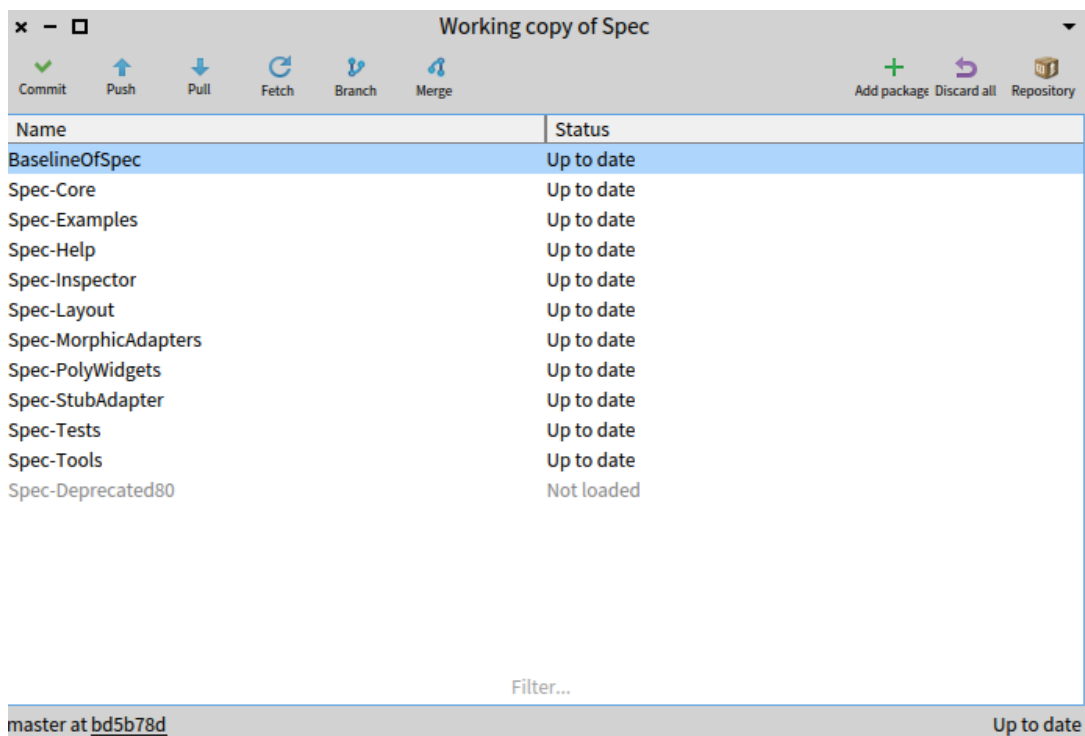


Slika 7

Sa leve strane Iceberg browser-a imate opcije za kreiranje novog repozitorijuma, dok se sa desne strane pojavljuje forma za unos potrebnih podataka. Možete kreirati novi lokalni repozitorijum, importovati postojeći klon sa vašeg računara, ili klonirati repozitorijum sa popularnih platformi kao što su GitHub, GitLab ili Bitbucket. Takođe, možete uneti direktan URL za kloniranje repozitorijuma sa drugih servera ako prethodne opcije ne odgovaraju. Ove opcije omogućavaju fleksibilnost u radu sa različitim repozitorijumima u Iceberg-u.

3.2 Dodavanje paketa

Ako je vaš repozitorijum nov, potrebno je dodati pakete. Možete koristiti postojeći paket ili kreirati novi Pharo paket. Zatim otvorite Working Copy Browser vašeg repozitorijuma dvoklikom na njega ili izborom opcije "Packages" u kontekst meniju. U ovom browser-u će se prikazati lista paketa u vašem repozitorijumu, koja će biti prazna ako je repozitorijum nov. Ovo vam omogućava pregled i upravljanje paketima u vašem projektu.

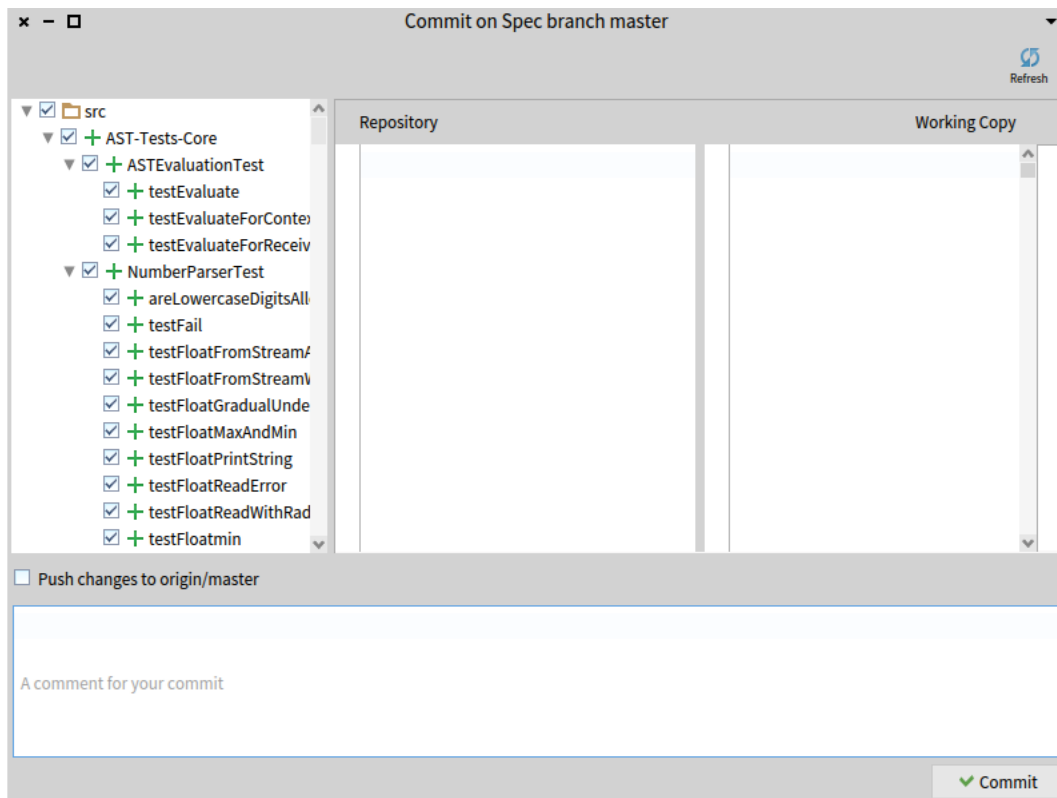


Slika 8

Nakon otvaranja Working Copy Browser-a, kliknite na dugme "Add package" da biste dodali sistemski paket u vaš repozitorijum. Odabrani paket će se pojaviti na listi praćenih paketa i biti označen kao promenjen (zeleni boja, zvezdica). Na ovaj način, paket postaje deo verzionisanog koda i može biti upravljan kroz Iceberg.

3.3 Commit

Kada je vaš repozitorijum označen kao izmenjen zbog promena u kodu, možete te promene sačuvati kroz commit. Da biste to uradili, otvorite Working Copy Browser i kliknite na dugme "Commit". Ovo će otvoriti commit dijalog gde možete uneti opis promena i sačuvati ih u repozitorijum. Ovaj proces omogućava praćenje verzija i lakšu kontrolu nad kodom u Iceberg-u.



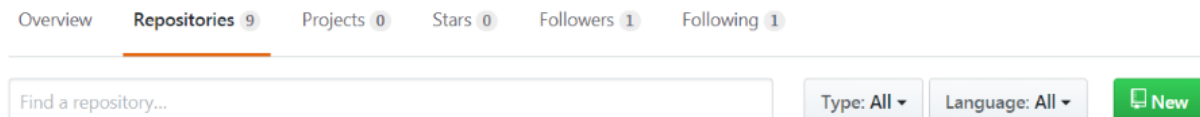
Slika 9

Na levoj strani vidite stablo koje prikazuje sve promene u paketima koji se prate u repozitorijumu. Možete selektovati definiciju sa leve strane i videti razlike u desnom panelu. Na dnu možete uneti poruku za commit.

Pored toga, iz kontekstualnog menija stabla možete poništiti promenu, pregledati promenjeni metod/klasu, ili selektovati šta želite da commit-ujete. Na kraju, kliknite na dugme 'Commit'.

4. Kreiranje repozitorijuma na GitHub-u

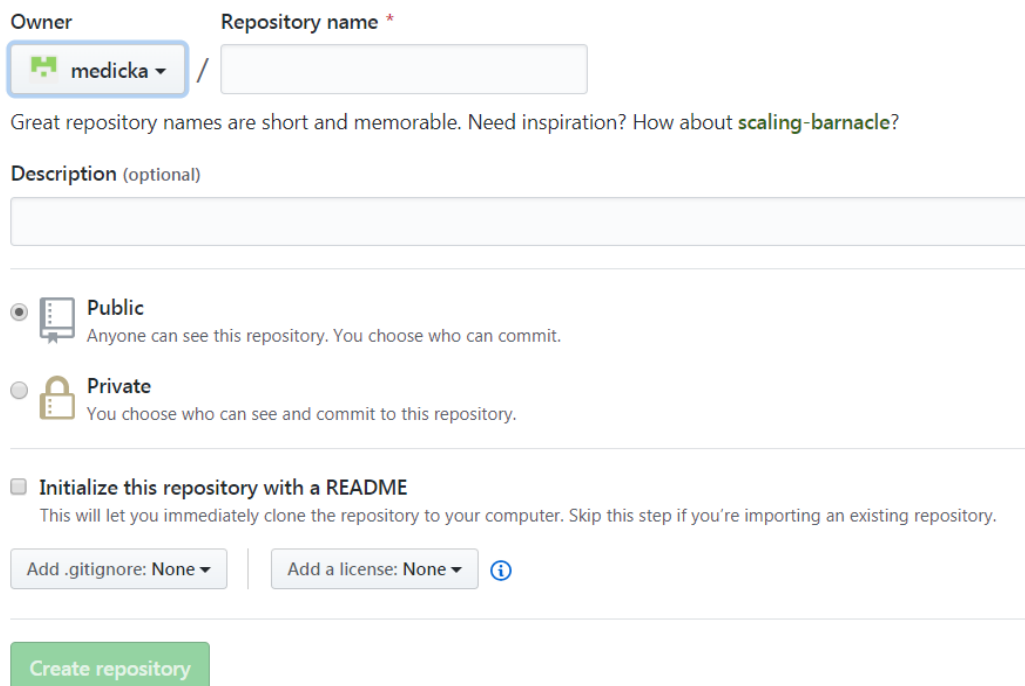
Da biste kreirali repozitorijum na GitHub-u, prvo posetite sajt i prijavite se (ako već imate nalog) ili se registrujte (da biste napravili novi nalog). Zatim idite na svoj profil i izaberite Repositories.



Slika 10

U gornjem desnom uglu biće dugme New; kliknite na njega da biste kreirali novi repozitorijum.

U novom prozoru unesite naziv, opis i README ako želite. Takođe možete odlučiti ko će videti vaš repozitorijum (Public - svi; Private - vi birate osobe koje će imati pristup). Nakon što popunite potrebne informacije, kliknite na dugme Create repository.

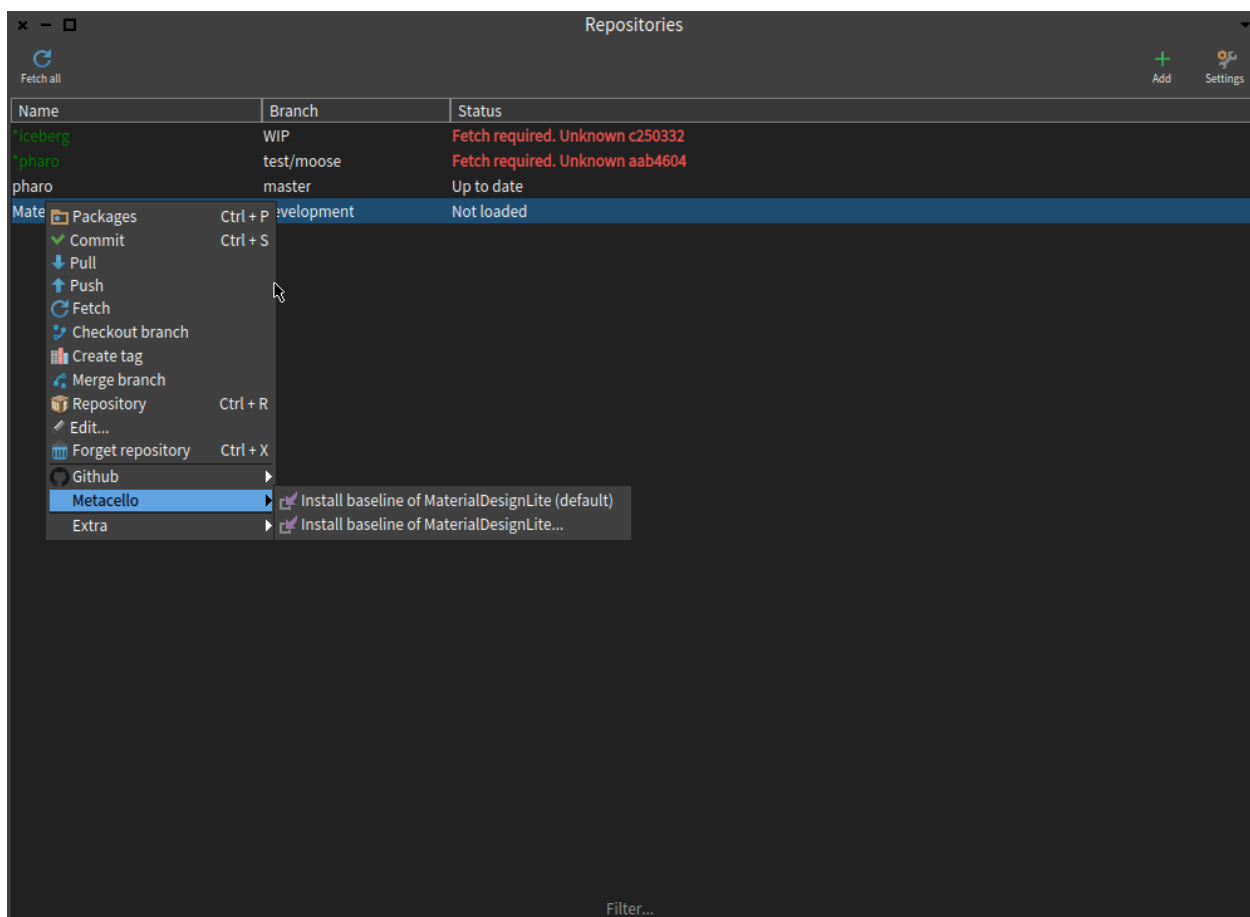


Slika 11

5. *Baseline, Iceberg i GitHub*

U Pharo 7, predstavljen je novi alat za upravljanje git repozitorijumima: Iceberg. Ovaj alat omogućava programerima da učitaju projekte preko korisničkog interfejsa.

Prvi korak je da dodate svoj git projekat u Iceberg. Zatim, kliknite desnim tasterom miša na naziv projekta kako biste pristupili Metacello pod-meniju i učitali projekat.



Slika 12

6. Zaključak

Iceberg je alat koji omogućava jednostavno upravljanje git repozitorijumima kroz korisnički interfejs, pružajući programerima mogućnost da lako prate i integrišu promene u kodu. Korišćenjem Iceberg-a, programeri mogu vizualizovati istoriju promena i upravljati verzijama na intuitivan način.

Baseline, s druge strane, automatizuje proces upravljanja zavisnostima i verzijama paketa unutar Pharo ekosistema. Ovaj alat osigurava da su svi paketi pravilno instalirani i ažurirani, što smanjuje rizik od problema sa kompatibilnošću i olakšava rad na projektima.

GitHub, kao centralna platforma za verzionisanje i kolaboraciju, omogućava skladištenje, deljenje i praćenje koda. Njegova integracija sa Iceberg-om i Baseline-om omogućava efikasno preuzimanje zavisnosti i praćenje promena, čime se poboljšava koordinacija i kvalitet razvoja softvera.

Sinergija između Iceberg-a, Baseline-a i GitHub-a doprinosi boljoj organizaciji i upravljanju softverskim projektima u Pharo okruženju. Ovi alati zajedno omogućavaju transparentnost, stabilnost i efikasnost u razvoju i održavanju kodova, što je ključno za uspeh u savremenom razvoju softvera.

7. Literatura

1. Pharo Wiki (<https://github.com/pharo-open-documentation/pharo-wiki/tree/master>)