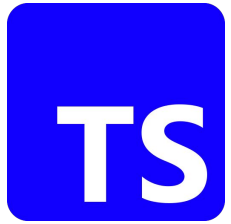
Code Academy



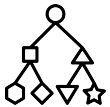
TypeScript CRUD aplikacijos kūrimas

TypeScript

12-15 paskaitos

Paskaitos eiga

Pritaikyti TypeScript temų žinias praktiniame darbe



Skaidyti kodą pagal atskiras atsakomybes

Tvarkingai struktūrizuoti projektą

Atlikti visus duomenų manipuliacijos veiksmus - CRUD

Projekto struktūrizavimas

Kaip struktūrizuoti projektą

Projekto struktūrizavimas priklauso nuo aplinkybių:

- Kalbos
- Technologijų
- Projekto paskirties
- Naudojamų programos dizaino technologijų
- Apimties ir t.t.

Kaip struktūrizuoti projektą

Kiekvieno projekto struktūrizavimo taisykles apsprendžia komandos architektas, dizaineris arba vyresnysis programuotojas.

Svarbiausia projekto struktūrizavimo taisyklė - vienodumas. Kad ir kokias struktūrizavimo, failų grupavimo taisykles taikote - tai turi būti dėsninga.

Struktūrizavimo taisyklių gairės

Kuomet kuriate taisykles pagal kurias bus stuktūrizuojamas projektas, laikykitės šių gairių:

- Jūsų projektas plėsis ir augs. Tai kas atrodo tvarkinga (laikinai) mažam projektui, nebus tvarkinga didesniam projektui
- Stenkitės atskirti struktūrizuojamas dalis, kad prie jų galėtų dirbti skirtingi programuotojai, nedirbant su tais pačias failais

Struktūrizavimo taisyklių gairės

- Struktūrizuojamos dalys turi būti:
  - Perpanaudojamos
  - Pasiekiamos.
  - Perkeliamos
  - Papildomos
  - Testuojamos
- Projekto aplankų pavadinimų medis turi atrodyti kaip knygos ar mokslinio darbo turinys

Gerosios programavimo praktikos

Projekto vystymo problemos

Pradedant kurti programą sunku nuspėti kaip keisis:

- Funkcionalumas
- Dydis
- Struktūra
- Paskirtis
- Duomenys
- Ryšiai su kitomis programomis

Projekto vystymo problemos

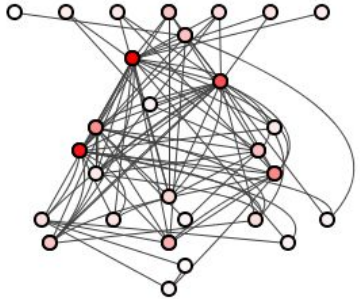
Programuotojų inžineriniai sprendimai pritaikyti mažam projektui, dažniausiai būna konkretūs ir skirti siauram panaudojimui.

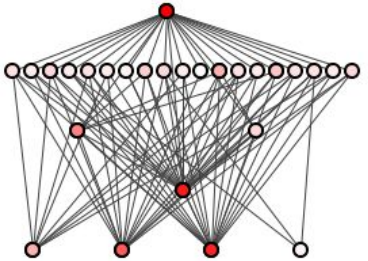
Plečiantis projekto apimčiai konkretūs (specifiniai) inžineriniai sprendimai sukausto esantį projektą nuo tolimesnio plėtojimo.

Projekto vystymo problemos

Naujo funkcionalumo diegimas reikalauja jau esamo projekto kodo perrašymo, kas užima nemažai laiko.

Kodas tampa sunkiai skaitomas, sunku integruoti naujus programuotojus į komandą ir sunku išlaikyti esamus programuotojus. Praktikoje tai vadinama *Spaghetti code*. Lietuvių įmonėse dar sakoma “primakaroninta kodo”





Dizaino šablonai ir tvarkinga architektūra

Patyrę programuotojai, susidūrę su neigiamais konkretaus kodo kūrimo padariniais, projektus pradeda kurti iš karto su mintimi kad jie plėsis.

Tam jie naudoja kodo struktūrizavima ir dizaino šablonus, tarsi projektas būtų didelis.

Dizaino šablonai ir tvarkinga architektūra

Nors sulėtėja pirminis projekto vystymo greitis, tai leidžia projektui gyvuoti ilgiau, dėl to:

- Projekto vystymo darbai aiškiai apibrėžiami
- Prie projekto gali dirbti daug žmonių
- Naujo funkcionalumo diegimas nekelia didelių problemų
- Lengva rasti darbuotojų ir išlaikyti senuosius
- Ilgalaikiai projektai pritraukia patyrusius programuotojus

S.O.L.I.D programavimo principai

- Single responsibility - kiekviena funkcija ar klasė yra skirta tik vienai paskirčiai. Plečiantis projektui tai didina skaitomumą ir supaprastina perpanaudojimą
- Open-closed - programavimo konstruktai turi būti baigtiniai ir nekeičiami, tačiau aprašyti tokiu būdu, kad jiems būtų galima nustatyti vykdomasias funkcijas ar aplinkybes
- Liskov’s substitution - Jeigu X klasės objektai turi savybę ar metodą, tai tuomet ir X paveldinčios klasės Y objektai privalo turėti savybę ir metodą tuo pačiu pavadinimu, skirtą tai pačiai paskirčiai

S.O.L.I.D programavimo principai

- Interface segregation - aprašant abstrakcijas naudojant interface’sus, geriau kurti mažus interface’sus. Taip išvengiama kad programuotojai yra priversti implementuoti abstrakčius metodus, kurių jiems nereikia.

- Dependency inversion principle - klasės turi priklausyti nuo abstrakcijų, o ne nuo konkrečių klasių. Tokiu būdu galima lanksčiai perduoti bet kokią abstrakcijos implementacija. Kodas tampa perpanaudojamesnis.

CRUD duomenų manipuliacijos veiksmai

CRUD projektas

CRUD projektas tai praktikoje naudojamas pavyzdinis projektas, kurio tikslas pademonstruoti pagrindinius manipuliacijos veiksmus:

- C - create
- R - read
- U - update
- D - delete

CRUD projektas tikslai

- Užtvirtinti įgytas TypeScript žinias
- Apjungti skirtingų TS temų žinias
- Susipažinti su naršyklės tipais
- Kurti struktūrizuotą ir plečiamą kodą
- Pritaikyti gerasias programavimo praktikas

Iki kito karto!