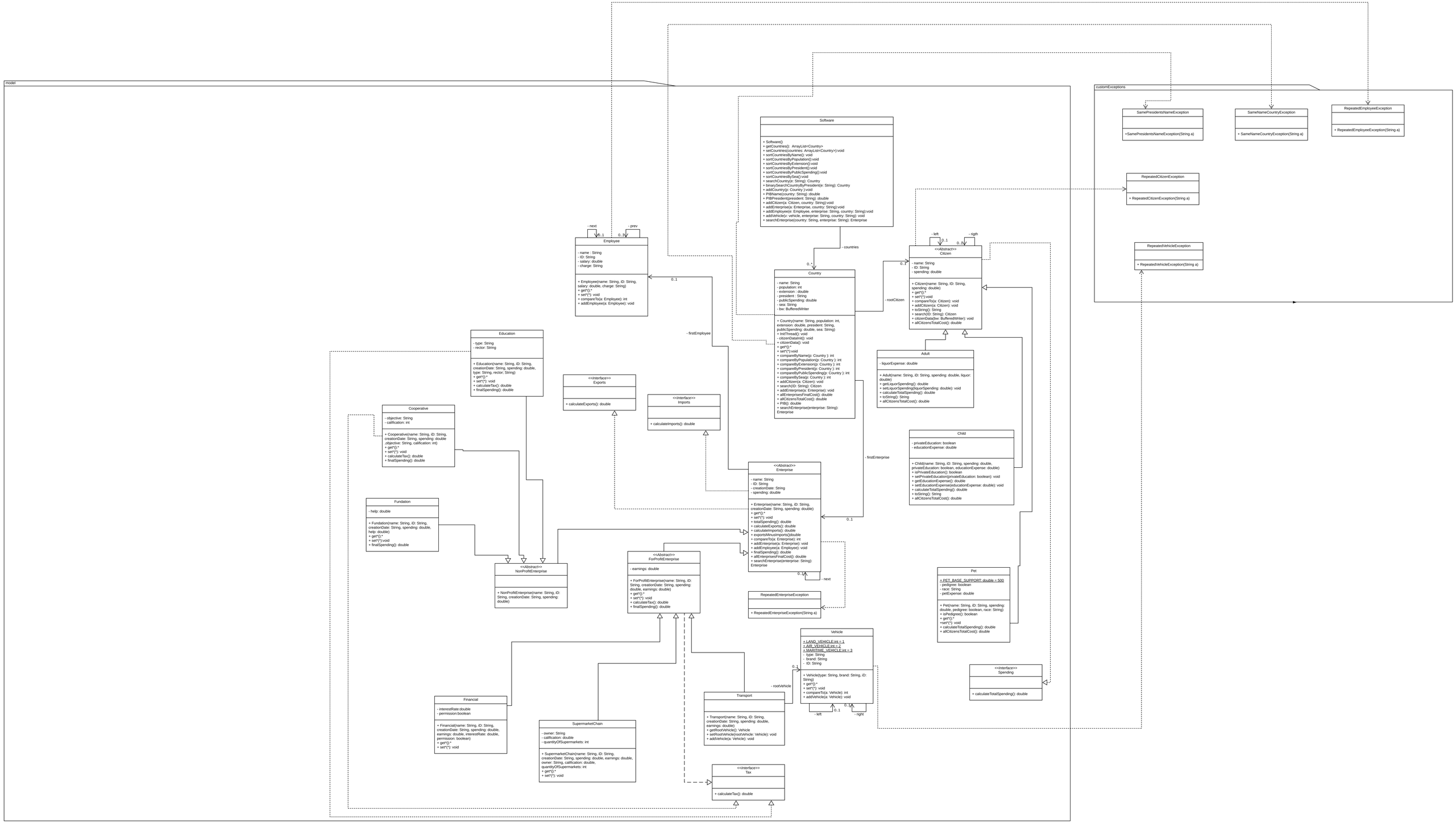
model
customExceptions
SamePresidentsNameException
+SamePresidentsNameException(String a)
SameNameCountryException
+ SameNameCountryException(String a)
RepeatedEmployeeException
+ RepeatedEmployeeException(String a)
RepeatedCitizenException
+ RepeatedCitizenException(String a)
RepeatedVehicleException
+ RepeatedVehicleException(String a)
Software
+ Software()
+ getCountries(): ArrayList<Country>
+ setCountries(countries: ArrayList<Country>):void
+ sortCountriesByName(): void
+ sortCountriesByPopulation():void
+ sortCountriesByExtension():void
+ sortCountriesByPresident():void
+ sortCountriesbyPublicSpending():void
+ sortCountriesbySea():void
+ searchCountry(e: String): Country
+ binarySearchCountryByPresident(e: String): Country
+ addCountry(p: Country):void
+ PIBName(country: String): double
+ PIBPresident(president: String): double
+ addCitizen(a: Citizen, country: String):void
+ addEnterprise(a: Enterprise, country: String):void
+ addEmployee(e: Employee, enterprise: String, country: String):void
+ addVehicle(v: vehicle, enterprise: String, country: String): void
+ searchEnterprise(country: String, enterprise: String): Enterprise
- countries
0..*
Country
- name: String
- population: int
- extension : double
- president : String
- publicSpending: double
- sea: String
- bw: BufferedWriter
+ Country(name: String, population: int, extension: double, president: String, publicSpending: double, sea: String)
+ initThread(): void
- citizenDataInit(): void
+ citizenData(): void
+ get(*):*
+ set(*):void
+ compareByName(p: Country): int
+ compareByPopulation(p: Country): int
+ compareByExtension(p: Country): int
+ compareByPresident(p: Country): int
+ compareByPublicSpending(p: Country): int
+ compareBySea(p: Country): int
+ addCitizen(a: Citizen): void
+ search(ID: String): Citizen
+ addEnterprise(a: Enterprise): void
+ allEnterprisesFinalCost(): double
+ allCitizensTotalCost(): double
+ PIB(): double
+ searchEnterprise(enterprise: String): Enterprise
- next
- prev
0..1
0..N
Employee
- name : String
- ID: String
- salary: double
- charge: String
+ Employee(name: String, ID: String, salary: double, charge: String)
+ get(*):*
+ set(*): void
+ compareTo(a: Employee): int
+ addEmployee(a: Employee): void
0..1
- firstEmployee
- left
- righ
0..1
0..N
<<Abstract>>
Citizen
- name: String
- ID: String
- spending: double
+ Citizen(name: String, ID: String, spending: double)
+ get(*):*
+ set(*):void
+ compareTo(a: Citizen): void
+ addCitizen(a: Citizen): void
+ toString(): String
+ search(ID: String): Citizen
+ citizenData(bw: BufferedWriter): void
+ allCitizensTotalCost(): double
0..1
- rootCitizen
Adult
- liquorExpense: double
+ Adult(name: String, ID: String, spending: double, liquor: double)
+ getLiquorSpending(): double
+ setLiquorSpending(liquorSpending: double): void
+ calculateTotalSpending(): double
+ toString(): String
+ allCitizensTotalCost(): double
Child
- privateEducation: boolean
- educationExpense: double
+ Child(name: String, ID: String, spending: double, privateEducation: boolean, educationExpense: double)
+ isPrivateEducation(): boolean
+ setPrivateEducation(privateEducation: boolean): void
+ getEducationExpense(): double
+ setEducationExpense(educationExpense: double): void
+ calculateTotalSpending(): double
+ toString(): String
+ allCitizensTotalCost(): double
- firstEnterprise
0..1
Education
- type: String
- rector: String
+ Education(name: String, ID: String, creationDate: String, spending: double, type: String, rector: String)
+ get(*):*
+ set(*): void
+ calculateTax(): double
+ finalSpending(): double
<<Interface>>
Exports
+ calculateExports(): double
<<Interface>>
Imports
+ calculateImports(): double
Cooperative
- objective: String
- calification: int
+ Cooperative(name: String, ID: String, creationDate: String, spending: double ,objective: String, calification: int)
+ get(*):*
+ set(*): void
+ calculateTax(): double
+ finalSpending(): double
Fundation
- help: double
+ Fundation(name: String, ID: String, creationDate: String, spending: double, help: double)
+ get(*):*
+ set(*):void
+ finalSpending(): double
<<Abstract>>
NonProfitEnterprise
+ NonProfitEnterprise(name: String, ID: String, creationDate: String, spending: double)
<<Abstract>>
ForProfitEnterprise
- earnings: double
+ ForProfitEnterprise(name: String, ID: String, creationDate: String, spending: double, earnings: double)
+ get(*):*
+ set(*): void
+ calculateTax(): double
+ finalSpending(): double
<<Abstract>>
Enterprise
- name: String
- ID: String
- creationDate: String
- spending: double
+ Enterprise(name: String, ID: String, creationDate: String, spending: double)
+ get(*):*
+ set(*): void
+ totalSpending(): double
+ calculateExports(): double
+ calculateImports(): double
+ exportsMinusImports():double
+ compareTo(a: Enterprise): int
+ addEnterprise(a: Enterprise): void
+ addEmployee(a: Employee): void
+ finalSpending(): double
+ allEnterprisesFinalCost(): double
+ searchEnterprise(enterprise: String): Enterprise
0..1
- next
RepeatedEnterpriseException
+ RepeatedEnterpriseException(String a)
Pet
+ PET_BASE_SUPPORT: double = 500
- pedigree: boolean
- race: String
- petExpense: double
+ Pet(name: String, ID: String, spending: double, pedigree: boolean, race: String)
+ isPedigree(): boolean
+ get(*):*
+set(*): void
+ calculateTotalSpending(): double
+ allCitizensTotalCost(): double
Vehicle
+ LAND_VEHICLE:int = 1
+ AIR_VEHICLE:int = 2
+ MARITIME_VEHICLE:int = 3
- type: String
- brand: String
- ID: String
+ Vehicle(type: String, brand: String, ID: String)
+ get(*):*
+ set(*): void
+ compareTo(a: Vehicle): int
+ addVehicle(a: Vehicle): void
0..1
- left
0..1
0..N
- right
<<Interface>>
Spending
+ calculateTotalSpending(): double
- rootVehicle
Financial
- interestRate:double
- permission:boolean
+ Financial(name: String, ID: String, creationDate: String, spending: double, earnings: double, interestRate: double, permission: boolean)
+ get(*):*
+ set(*): void
SupermarketChain
- owner: String
- calification: double
- quantityOfSupermarkets: int
+ SupermarketChain(name: String, ID: String, creationDate: String, spending: double, earnings: double, owner: String, calification: double, quantityOfSupermarkets: int
+ get(*):*
+ set(*): void
Transport
+ Transport(name: String, ID: String, creationDate: String, spending: double, earnings: double)
+ getRootVehicle(): Vehicle
+ setRootVehicle(rootVehicle: Vehicle): void
+ addVehicle(a: Vehicle): void
<<Interface>>
Tax
+ calculateTax(): double