# Scenario

Třída Scenario představuje správce objektů, které implementují rozhraní INaboj, a umožňuje jejich přidávání, odebírání, získávání jejich pozic a kreslení celého scénáře na grafickém plátně.

## Atributy:

* INaboj?[] charges: Pole, které uchovává objekty typu INaboj. Inicializováno s kapacitou 1 a dynamicky rozšiřováno.
* int freeIndex: Index první volné pozice v poli charges.
* int chargesCount: Aktuální počet přidaných objektů v poli.

## Metody:

* **GetCharges**
  + Vrací aktuální pole charges obsahující objekty typu INaboj.
* **AddCharge(INaboj naboj)**
  + Přidává nový náboj do pole charges. Pokud je pole plné, jeho kapacita se zdvojnásobí.
* **RemoveCharge(INaboj naboj)**
  + Odebere náboj z pole na základě předaného objektu naboj.
* **RemoveCharge(int id)**
  + Odebere náboj z pole na základě jeho ID.
* **GetPositions**
  + Vrací souřadnice všech nábojů ve formě dvou polí (X a Y). Každý náboj vrací 2 pozice (přední a zadní okraj).
* **Draw(Graphics g, float width, float height, int startTime)**
  + Vypočítá výřez k zobrazení tak, že vezme nejmenší obdélník obsahující všechny náboje, rozšíří ho ve všech směrech o 25 %, zvětší, roztáhne, aby v jednom směru zaplnil okno, nakonec doplní výřez ve druhém směru
  + Postupně vykreslí pozadí, osy, náboje a nakonec sondu

Třída poskytuje základní funkce pro správu a vizualizaci nabitých částic (INaboj), přičemž dynamicky upravuje jejich zobrazení na základě jejich vlastností a pozic.

# Grid

Třída Grid implementuje rozhraní IGrid a zajišťuje vykreslení mřížky s osami v grafickém prostředí. Tato třída je navržena tak, aby kreslila mřížku s šipkami na osách.

## Metody:

* **GetNumberOfGridLines**
  + Zatím není implementována. Bude vracet počet čar v mřížce (zřejmě podle os X a Y).
* **SetNumberOfGridLines(int row, int column)**
  + Zatím není implementována. Bude nastavovat počet čar v mřížce pro zadané počty řádků a sloupců.
* **Draw(Graphics g, PointF topLeft, PointF bottomRight, Pen pen, Brush brush, float tipLength)**
  + Veřejná metoda, která vykreslí mřížku mezi body topLeft (levý horní roh) a bottomRight (pravý dolní roh).
  + Používá interní metodu DrawGrid pro kreslení samotných os a šipek na konci os.

## Privátní metody:

* **DrawArrows(Graphics g, Brush brush, PointF rightCenter, PointF topCenter, float tipLength)**
  + Kreslí šipky na koncích os.
  + rightCenter určuje pozici šipky na pravé ose X a topCenter určuje pozici šipky na horní ose Y.
  + tipLength definuje délku špiček šipek.
* **DrawGrid(Graphics g, Pen pen, Brush brush, PointF topLeft, PointF bottomRight, float tipLength)**
  + Kreslí samotnou mřížku.
  + Počítá střední body os X a Y a kreslí je pomocí zadaného pera pen. Šipky jsou vykresleny pomocí metody DrawArrows.

Třída je zodpovědná za kreslení základní mřížky s osami a šipkami, ale funkce pro práci s počtem čar v mřížce zatím není implementována.

# Naboj

Třída Naboj implementuje rozhraní INaboj a reprezentuje nabitou částici s určitým nábojem, polohou, poloměrem a unikátním ID. Tato třída také obsahuje metodu pro vykreslení nabité částice na grafickém plátně.

## Atributy:

* **float charge**: Hodnota elektrického náboje částice.
* **float radius**: Poloměr částice, inicializován na hodnotu 1f.
* **PointF center**: Souřadnice středu částice. Souřadnice Y je invertována při vytvoření objektu.
* **int id**: Unikátní identifikátor částice.

## Konstruktory:

* **Naboj(int charge, PointF center, int id)**: Konstruktor, který nastaví náboj, polohu a ID částice. Poloměr je implicitně nastaven na hodnotu 1f.

## Metody:

* **float GetCharge()**
  + Vrací aktuální hodnotu náboje částice.
* **void SetCharge(float charge)**
  + Nastaví hodnotu náboje částice.
* **PointF GetPosition()**
  + Vrací aktuální pozici (střed) částice.
* **void SetPosition(PointF point)**
  + Nastaví pozici částice.
* **float GetRadius()**
  + Vrací aktuální hodnotu poloměru částice.
* **void SetRadius(float radius)**
  + Nastaví nový poloměr částice.
* **int GetID()**
  + Vrací unikátní identifikátor částice.
* **void Draw(Graphics g, PointF panelCenter, float scale)**
  + Vykreslí částici na plátno Graphics g.
  + Vytvoří kruh pomocí barvy DarkTurquoise s aktuálním poloměrem částice.
  + Zobrazí popisek s hodnotou náboje částice uprostřed kruhu.
  + Pro vykreslení popisku je použito písmo, které se mění podle zadaného měřítka (scale).

Třída Naboj poskytuje základní funkce pro práci s nabitou částicí, jako je získávání a nastavování hodnot náboje, pozice, poloměru a unikátního ID, a také možnost vykreslení částice na grafické plátno.

# Probe

Třída Probe slouží k vizualizaci zkoumané částice (sondy) a jejího chování v elektrickém poli. Třída simuluje pohyb částice a vykresluje šipku znázorňující sílu elektrického pole v daném bodě.

## Atributy:

* **PointF center**: Střed sondy, reprezentovaný souřadnicemi.
* **float radius**: Poloměr kruhu, ve kterém sonda rotuje. Defaultní hodnota je 1f.
* **float anglePerSecond**: Úhel, který sonda urazí za sekundu. Defaultně je nastaven na π / 6 radiánů za sekundu.

## Konstruktor:

* **Probe(PointF center, float radius = 1f, float anglePerSecond = MathF.PI / 6)**
  + Inicializuje sondu se zadanými parametry (střed, poloměr rotace a úhel za sekundu).

## Metody:

* **void Draw(Graphics g, int startTime, INaboj[] charges, float scale)**
  + Vykresluje sondu na grafické plátno g.
  + Vypočítá aktuální polohu sondy podle času (startTime) a rychlosti rotace (anglePerSecond).
  + Simuluje vliv okolních nábojů (charges) na elektrické pole v daném bodě.
  + Vypočítá sílu elektrického pole a vykreslí vektor síly jako šipku.
  + Šipka je doplněna popiskem ukazujícím velikost síly elektrického pole v jednotkách TN/C.
* **private void DrawArrow(Graphics g, Vector2 sum, float scale, Color color)**
  + Vykresluje šipku reprezentující směr a velikost vektoru elektrického pole.
  + Šipka je vypočítána na základě výsledného vektoru pole sum a její velikost je přizpůsobena měřítku (scale).
  + Šipka má definovanou délku hrotu (tipLen), která se přizpůsobuje velikosti šipky a škálování.

## Výpočet vektoru elektrického pole:

* Sonda vypočítává elektrické pole způsobené náboji v okolí pomocí vzorce Coulombova zákona:
  + **E = k \* Q / r²**: kde E je elektrická intenzita, Q je náboj a r je vzdálenost mezi nábojem a sondou.
  + Součet vektorů pole pro všechny náboje je uložen v proměnné sum.

## Další informace:

* **Vykreslování textu a sondy**: Při vykreslování sondy se zobrazí také kruh s textem znázorňujícím sílu elektrického pole v daném bodě.
* **Transformace souřadnic**: Pro vykreslení se využívá transformace souřadnic (g.TranslateTransform), aby se pozice a měřítko přizpůsobily grafickému prostředí.

Tato třída umožňuje grafickou vizualizaci pohybu sondy a změny elektrického pole způsobené okolními náboji.