# Prototype 💻

Análise e Desenvolvimento de Sistemas - 6º Período

# Cenário

- Vimos no proxy uma maneira de **resolver a inicialização de objetos pesados**
- Usamos uma estratégia **on-demmand**
- E se for necessário criar **cópias** desses objetos pesados?
- E se for necessário **cópias de vários objetos diferente, independente do custo em memória**?

# Cenário

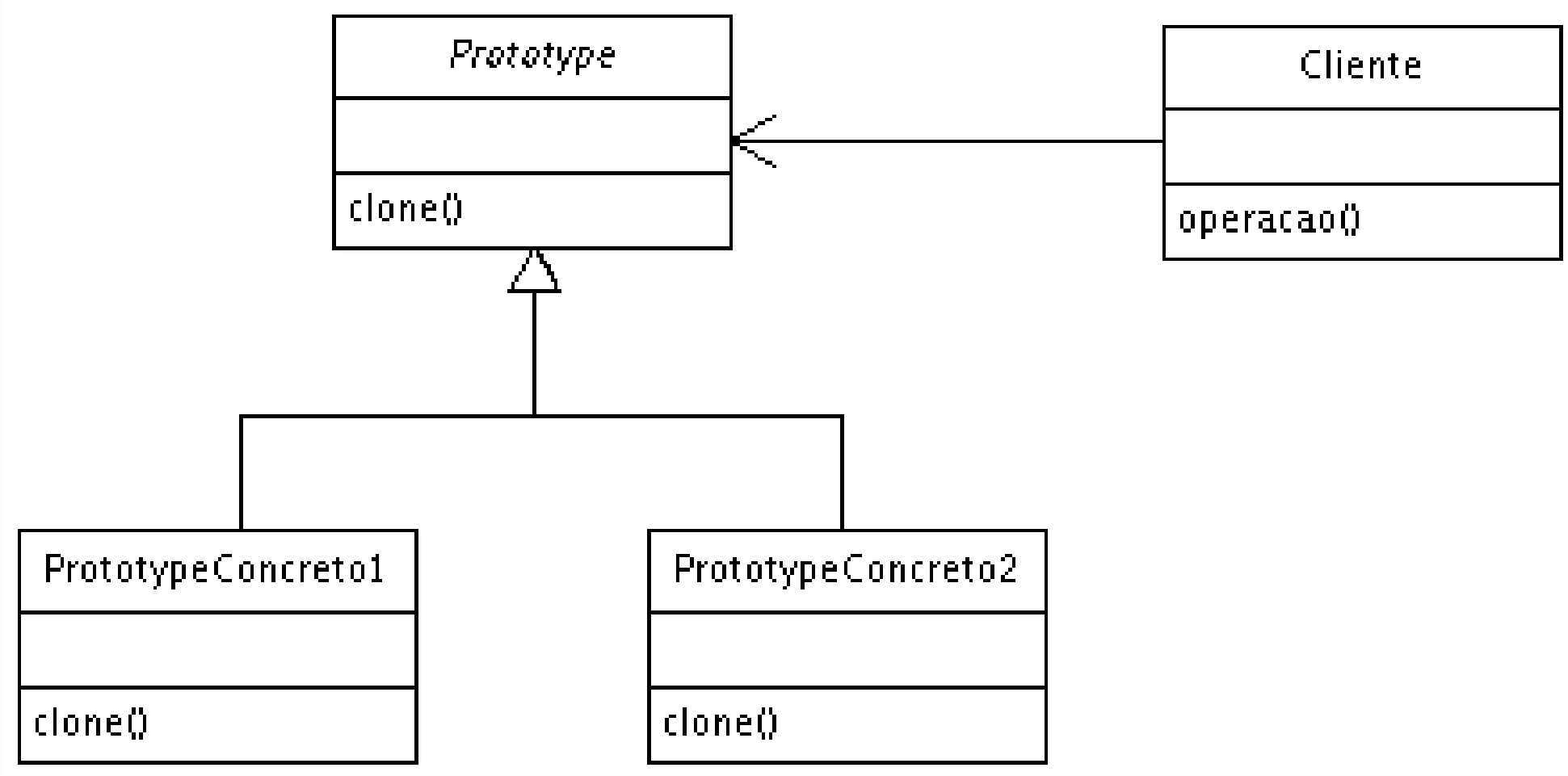
- Imagine que queremos implementar uma função no Portal para exportar relatórios
- Cada aluno pode gerar relatórios de vários tipos: Notas, Faltas, Atividades etc
- Dependendo do formato do relatório, pode ser **caro inicializar o objeto**
- Além disso, o **número de requisições** pode ser grande

# Prototype

- Com o Prorotype podemos ter **apenas um objeto instanciado que servirá como protótipo para outros**
- Os outros objetos serão **cópias** desses protótipo
- Isso permite que, nesse caso, **apenas uma inicialização seja feita**

# Prototype

- Essa abordagem **elimina a necessidade de criação de subclasses**
- Usamos um **registro dos protótipos**
- O client tem **acesso apenas a esse registro e não conhece a classe que implementa o objeto**


Prototype
clone()
Cliente
operacao()
PrototypeConcreto1
clone()
PrototypeConcreto2
clone()

# Prototype vs Proxy

- A **motivação** muda
- Perceba que o diagrama de classe entre os dois padrões é o mesmo
- No entanto, o **Proxy** nós entrega a capacidade de **controle de acesso**
- Já o **Protoype**, a inicialização de apenas um objeto