GeekBrains

Объектно-ориентированное программирование

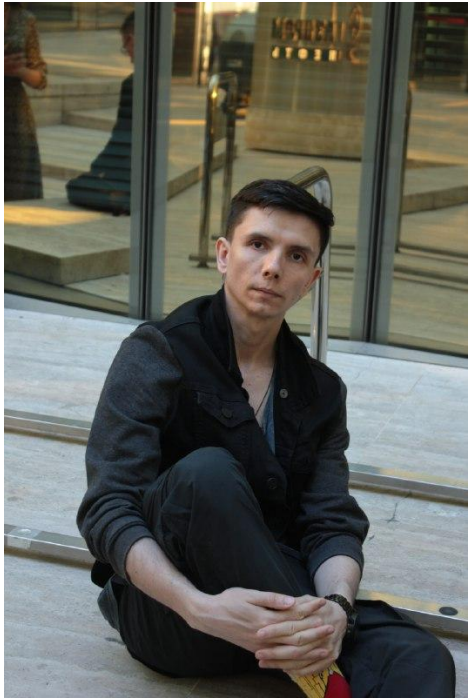
Семинар 1

Принципы ООП. Знакомство

Меня зовут Александр. Я сменил профессию в 33 года, закончил факультет Java-разработки на платформе “Geek Brains”.

Еще не закончив курс, нашёл свою первую работу и уже около 2 лет занимаюсь коммерческой разработкой.

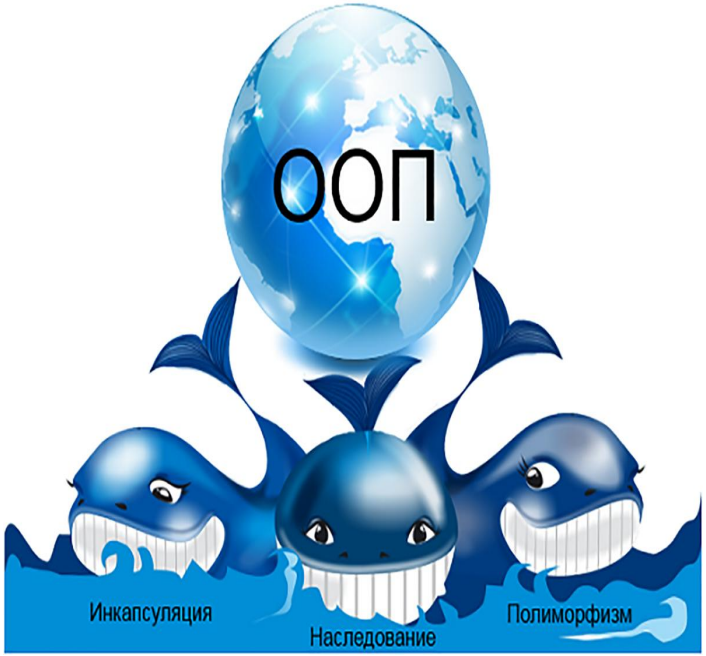
Разрабатывал навигационные приложения для гражданских и военных судов. На данный момент вхожу в команду backend-разработчиков одного из активно развивающихся стартапов.



Ваши вопросы

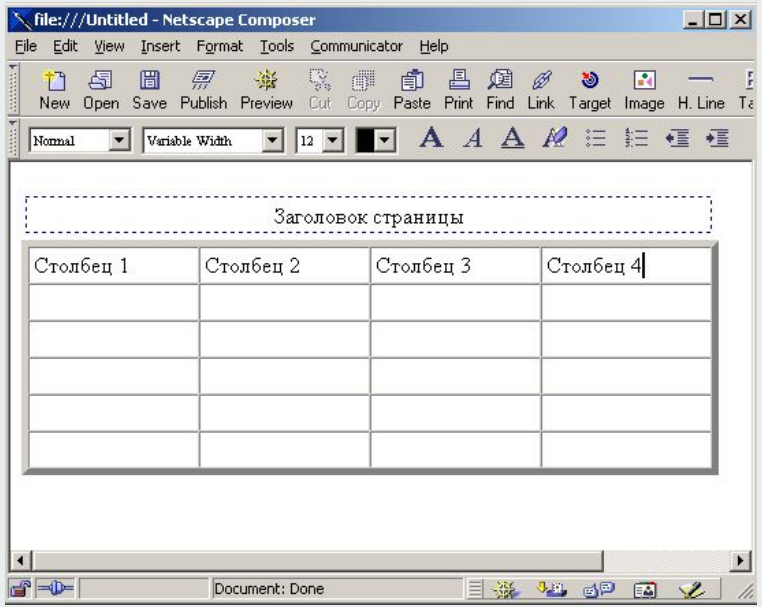
Сегодня на занятии:

1. Попытаемся усвоить и закрепить основные принципы концепции ООП:
   - абстракция;
   - наследование;
   - полиморфизм;
   - инкапсуляция;
2. В процессе занятия попытаемся ответить на несколько вопросов, которые часто задают на собеседовании на позицию junior-разработчика.



Принципы ООП

Абстракция как таблица с незаполненными колонками



Создаём абстракцию (СОСТОЯНИЕ):

1. Попробуйте описать животное, как объект из реальной жизни, но МАКСИМАЛЬНО без конкретных деталей;
2. Создайте final переменную TYPE, отражающую название класса;
3. Создайте класс **Animal** с полями: имя(кличка), окрас(цвет) и количество лап. Подумайте, к каким полям можно ограничить доступ;
   - тут попытаемся усвоить понятия объект, его экземпляр, ключевое слово **this,** как в java вызвать конструктор и что это такое, как управлять доступом к полям объекта;
   - попытаемся понять, для чего могут быть нужны несколько конструкторов и как организовать цепочку их вызова.
4. Выделим понятие СОСТОЯНИЯ объекта.

Продолжаем создание абстракции(ПОВЕДЕНИЕ):

1. Создайте в классе Animal ничего не возвращающие методы: проснуться(wakeUp), найти еду (findFood), поесть(eat), поиграть(toPlay), уснуть(goToSleep), говорить(speak);
2. Попробуйте придумать для этих методов максимально абстрактные реализации;
- какие проблемы вы возможно уже увидели?
3. Попробуйте дать определение ПОВЕДЕНИЯ объекта.
4. Ваши вопросы по концепции **АБСТРАКЦИЯ.**

Принципы ООП

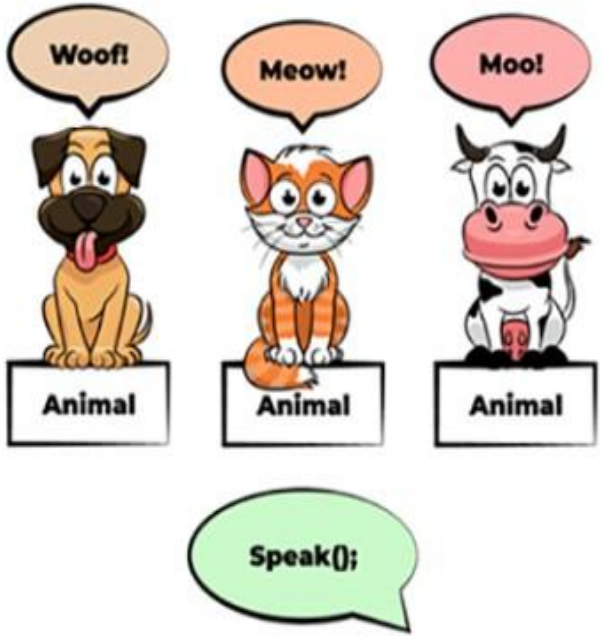
Наследование как дерево и его ветви с листьями.

Наследование. Класс Object:

1. Реальные вопросы из собеседования:
   - полностью ли java является ООП-языком? Почему?
   - сколько методов у класса Object? Сколько Вы можете назвать?
2. Удалите из класса Animal поле TYPE, а его функционал перенесите в метод getType
   - Подумайте, как метод getType() можно реализовать через методы класса Object.
   - Какое определение наследования Вы бы сейчас могли озвучить?
3. Создайте класс Cat, наследник Animal, создайте у Cat конструктор с двумя аргументами - name, color.
   - ключевое слово super;
4. Создайте по экземпляру классов Animal и Cat и вызовите у них метод getType()
   - дайте определения понятиям **объект класса и экземпляр класса**

Принципы ООП

Полиморфизм или я тоже так умею



Полиморфизм и @Override

1. Вызовите метод speak() у экземпляров классов Cat и Animal. Переопределите метод speak() у объекта Cat;
2. Попробуйте вывести на консоль экземпляр Cat. Как вы думаете, какой метод был вызван?
3. Создайте классы Dog, Duck с конструктором, имеющим параметры name и color. Переопределите метод speak() у этих объектов;
4. В Main-классе приложения создайте List<Animal> animals и наполните его экземплярами классов Cat, Dog и Duck. Обратитесь к каждому экземпляру в списке и вызовите у него метод speak();
5. В классе Duck создайте метод fly(). Снова обратитесь к каждому экземпляру в списке и попробуйте вызвать у него метод fly(). Ознакомьтесь с сообщением от компилятора. Какая проблема возникла при выполнении кода?

- оператор instanceof

6. Попробуйте дать определение **полиморфизма**
7. Вопрос из собеседования: чем отличаются overload и override?
8. Ваши вопросы.

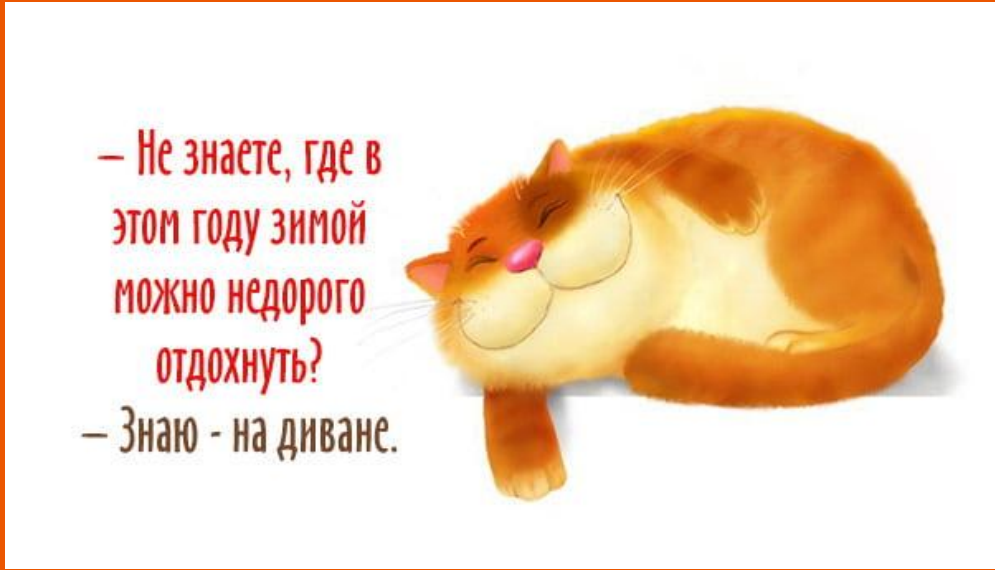
Принципы ООП

Инкапсуляция или, а... аа... а как это сделано?

Инкапсуляция это сокрытие?

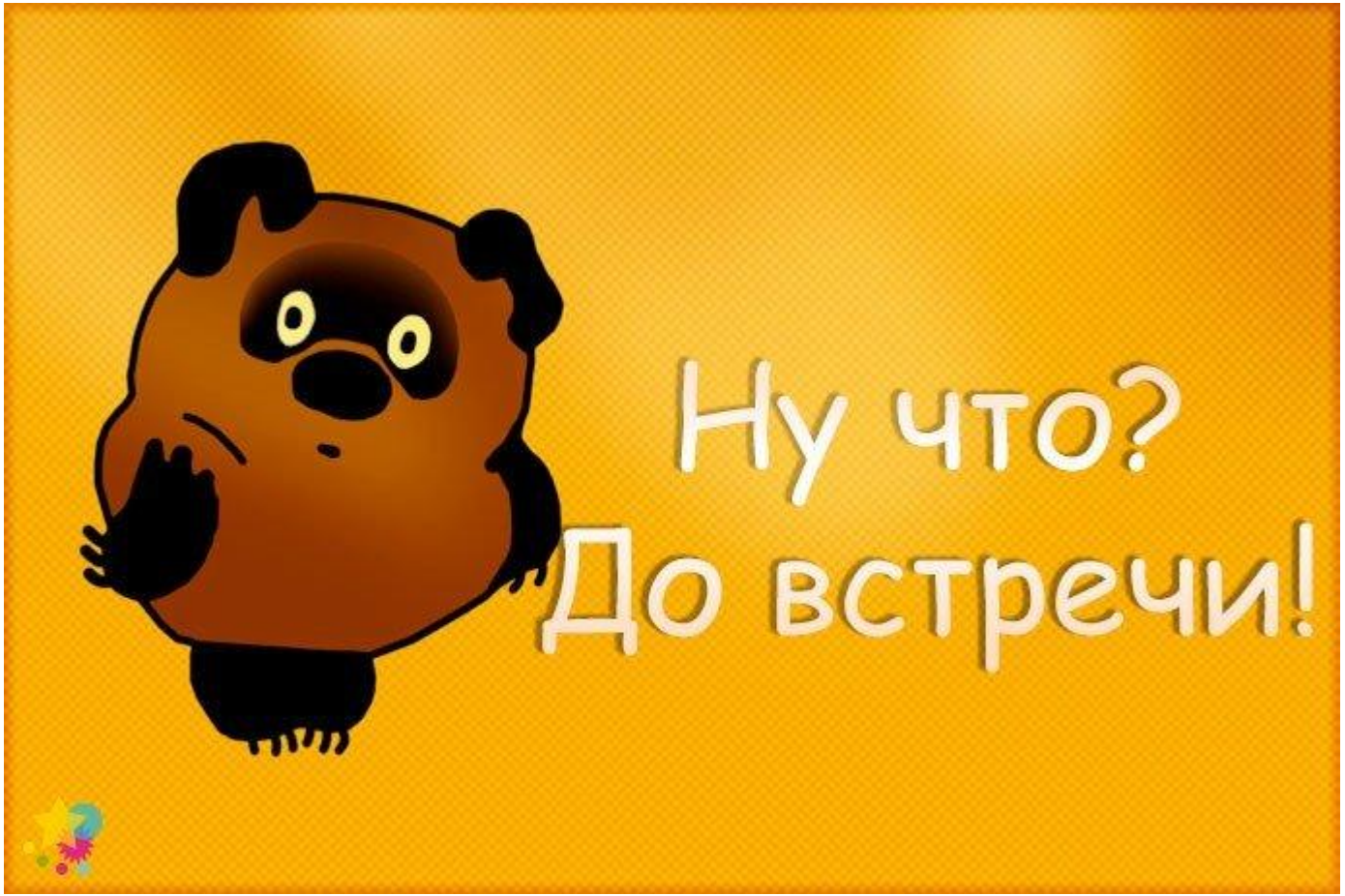
1. Скопируйте и перенесите поведенческие методы(кроме getType и speak) из класса Animal в объект Cat и переопределите их. Эти методы в классе Animal удалите.
   - модификаторы доступа;
   - как вы думаете, какие возникнут проблемы, если мы захотим заставить наших животных поохотиться?
2. Создайте в классе Animal метод hunt(), можете оставить его нереализованным. Какую проблему вы видите в таком решении?
3. Переопределите метод hunt() в объекте Cat, причем постарайтесь реализовать его, используя уже существующие поведенческие методы объекта. Как вы думаете, чем хорошо такое решение?
4. Вопрос на собеседовании: Как вы понимаете инкапсуляцию?

Как Ваше настроение?



Ваши вопросы и пожелания :)

1. Было ли сложно?
2. Получилось ли узнать/закрепить что-то новое, полезное?
3. Что не понравилось на занятии?
4. А что понравилось?
5. Что можно улучшить? Есть ли у Вас предложения?
6. Буду рад любым отзывам под записью занятия, это мотивирует! :)


Ну что?
До встречи!