

Modul I. Lesson 13

Flutter Development

Ravshan Anarqulov 02.22

Lesson Plan

- Repeat the previous lesson!
- Constructors
- Summary
- Tasks
- Answer questions!

Constructors

- Constructorlar - bu sinf instance larini yaratadigan yoki tuzadigan method lardir. Ya'ni, constructor lar yangi ob'ektlarni qurishadi. Constructorlar class bilan bir xil nomga ega va Constructor methodining yashirin qaytish turi ham class ning o'zi bilan bir xil.
- Constructor – bu class dan object yaratayotganda birinchi bo'lib ishga tushadigan method hisoblanadi. Bu sizga class dagi boshqa methodlar ishlashidan oldin biron ish bajarishingizga imkon beradi.
- Constructor nomi class nomi bilan bir xil bo'lishi kerak.
- Constructor hech narsa qaytarmaydi.
- Constructor o'z hayoti davomida faqat bir marta chaqiriladi, ya'ni Ob'ekt yaratilganda.

Default Constructor

- Parametrlarsiz yozilgan constructor default constructor deyiladi.
- Agar biz uni code da elon qilmasak dart compilyatori uni aftomatik qo'shib ketadi.
- C++ dan farqli o'laroq dart da destructor tushunchasi mavjud emas.

Default Constructor



A code editor window titled 'main.dart' with a dark background. It contains the following Dart code:

```
class Address {  
  var value = '';  
}
```

=



A code editor window titled 'main.dart' with a dark background. It contains the following Dart code, which is equivalent to the one on the left but includes an explicit default constructor:

```
class Address {  
  var value = '';  
  
  Address();  
}
```

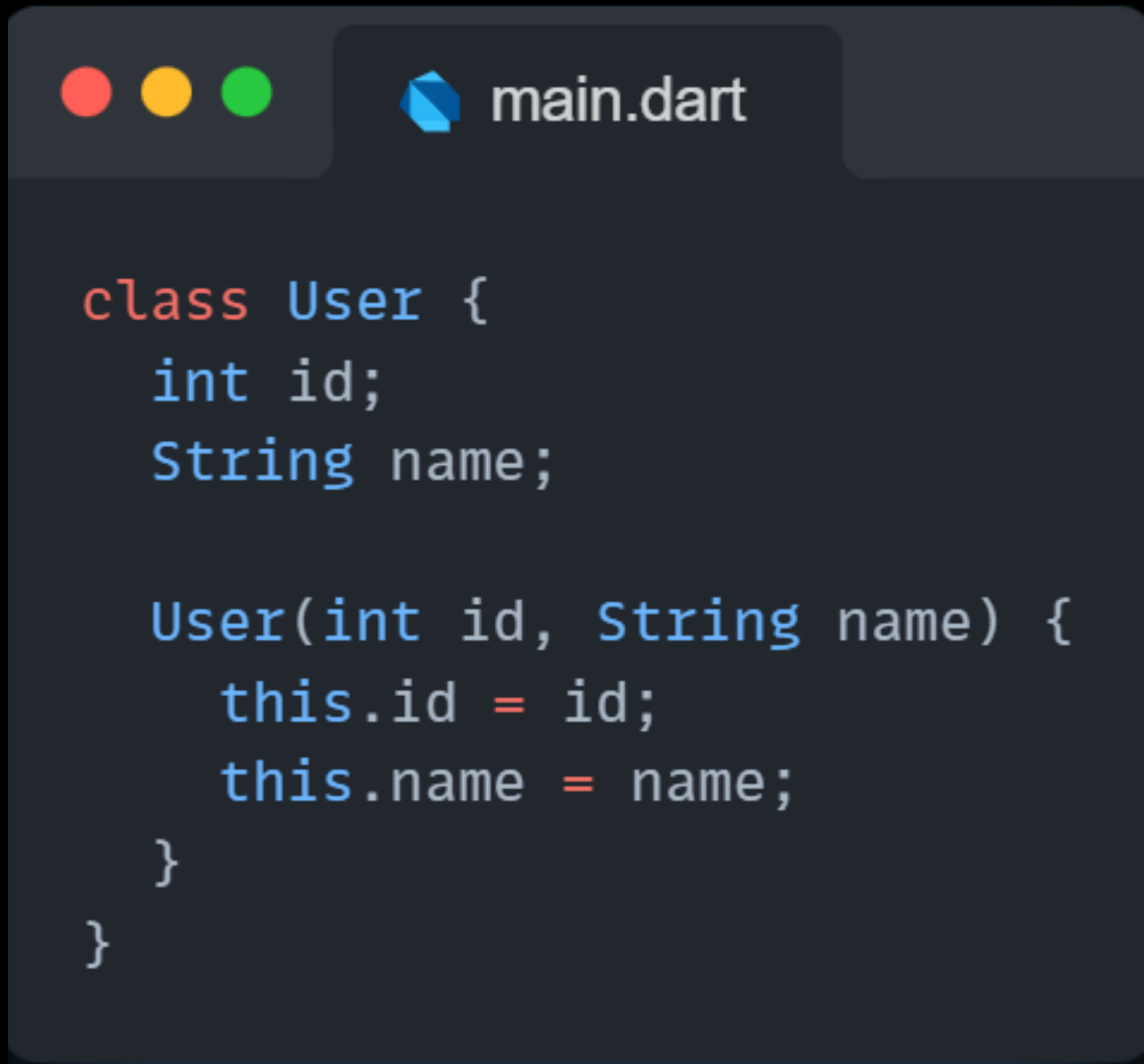
Exercise

- User nomli class yarating unda “int id”, “ String name” field lari va default constructori bo’lsin.

Generativ Constructors, Parameterized Constructor

- Agar siz constructor ga parametrlar beriladigan bo'lsa bunday constructor paramerli constructor deyiladi.
- Ushbu turdagi constructor odatda instance field lariga boshlang'ich qiymat bermoqchi bo'lganingizda ishlatiladi. Ushbu constructor ga bitta yoki bir nechta argumentlarni yuborishingiz mumkin.

- Long-form constructor



```
class User {  
  int id;  
  String name;  
  
  User(int id, String name) {  
    this.id = id;  
    this.name = name;  
  }  
}
```

- Short-form constructor

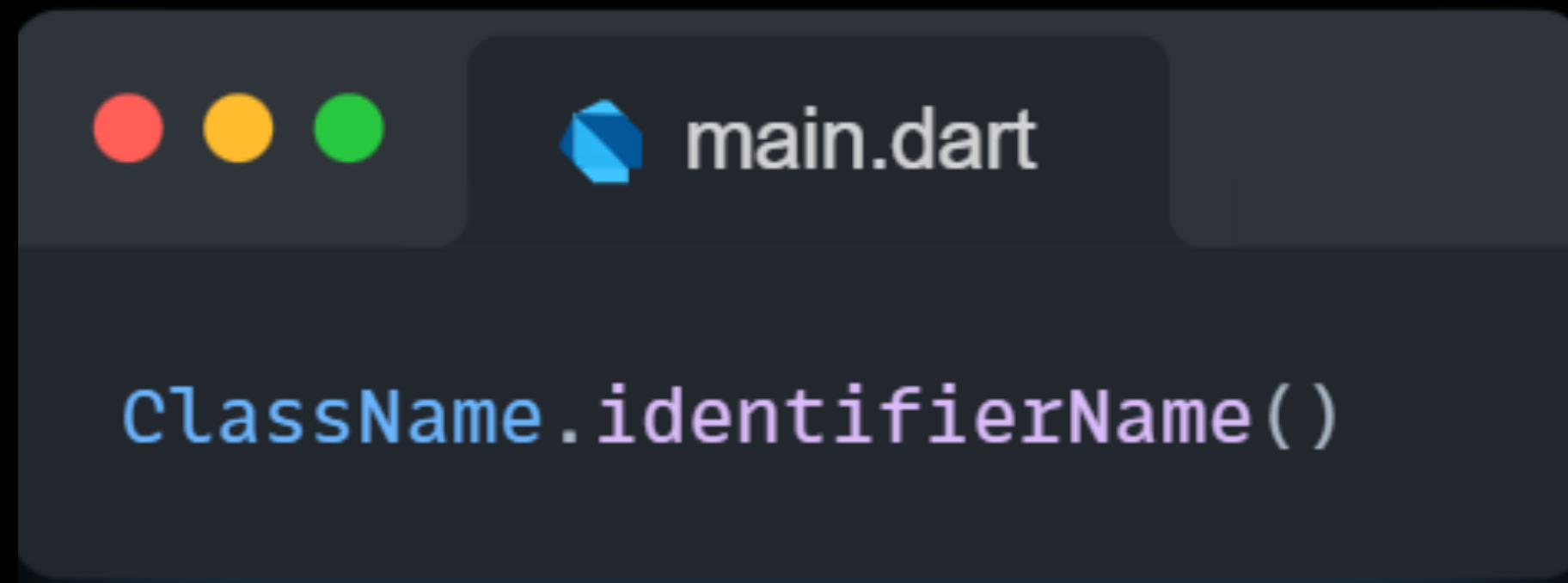


```
class User {  
  int id;  
  String name;  
  
  User(this.id, this.name);  
}
```

- Bu yerda this kalit so'zi joriy instance ga ishora qiladi.
- Buni faqat nom ziddiyati mavjud bo'lganda foydalaning. Aks holda, Dart this keyword ini inobatga olmaydi.

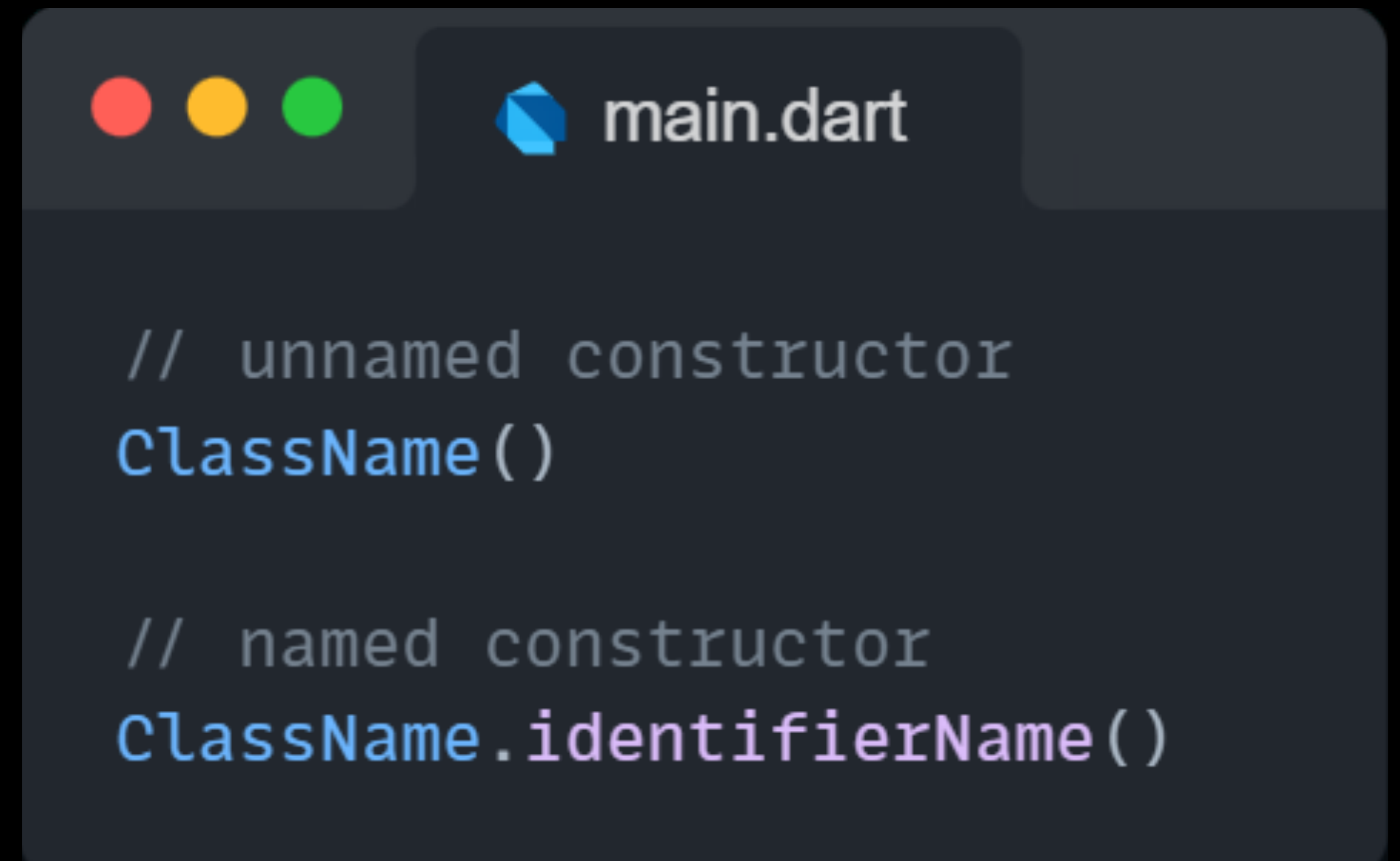
Named constructors

- Dart shuningdek, nomli konstruktor deb ataladigan konstruktorga ega, siz uni sinf nomiga identifikator qo'shish orqali yaratasisiz.
- Hozirgacha yaratgan constructor larimiz unnamed constructor lardir.



A code editor window titled 'main.dart' with a Dart logo icon. It contains the following code:

```
ClassName.identifierName()
```



A code editor window titled 'main.dart' with a Dart logo icon. It contains the following code:

```
// unnamed constructor  
ClassName()  
  
// named constructor  
ClassName.identifierName()
```

- Ba'zan turli funktsiyalarni bajarish uchun bizga bir nechta konstruktor kerak bo'ladi. Lekin bir xil nomli bir nechta konstruktor yarata olmaysiz.
- Ushbu muammoni bartaraf etish uchun dart da turli nomlar bilan bir nechta constructor lar yaratish imkoni mavjud.



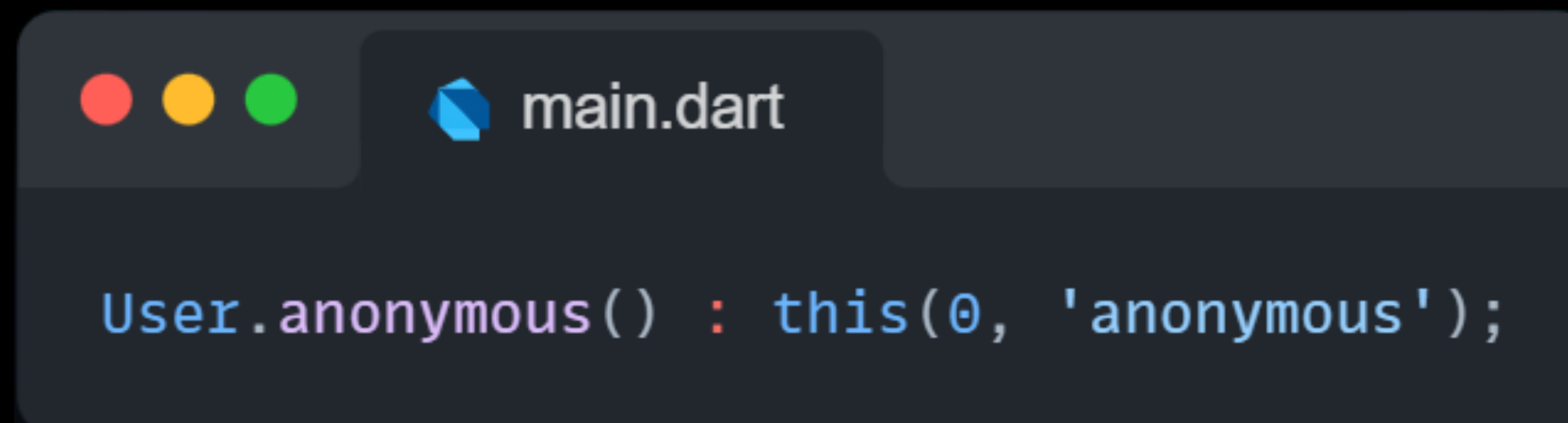
```
void main() {  
  final anonymousUser = User.anonymous();  
  print(anonymousUser);  
}  
  
class User {  
  int id = 0;  
  String name = '';  
  
  User(this.id, this.name);  
  
  User.anonymous() {  
    id = 0;  
    name = 'anonymous';  
  }  
}
```

Exercise

- `User.anonymous()` constructori orqali class instancini yarating, parameterized constructor orqali ham anonymous dagi bilan bir xil qiymatga ega bo'lgan instance yarating.

Redirect constructors

- Oldingi misolda ko'rganingizdek ikkala constructor ham o'zi fieldlarga qiymat biriktirmoqda.
- Bu codening qayta yozilayotganini bildiradi.
- Ushbu muammoni hal qilishning bir usuli - asosiy konstruktorni nomlangan konstruktordan chaqirishdir. Bu yo'naltirish yoki qayta yo'naltirish deb ataladi. Buning uchun yana this kalit so'zidan foydalanasiz



```
User.anonymous() : this(0, 'anonymous');
```

- Quyida ko'rib turganingizdek this keyword i orqali qiymat biriktirishni asosiy constructorga yo'nartirib yubarayabmiz va endi bizga field lardagi default qiymatlar kerakemas.



```
class User {  
  int id;  
  String name;  
  
  User(this.id, this.name);  
  
  User.anonymous() : this(0, 'anonymous');  
}
```

- Huddi funksiyalardadidek constructorlarda ham parametlarni turli shakillarda berish mumkun.

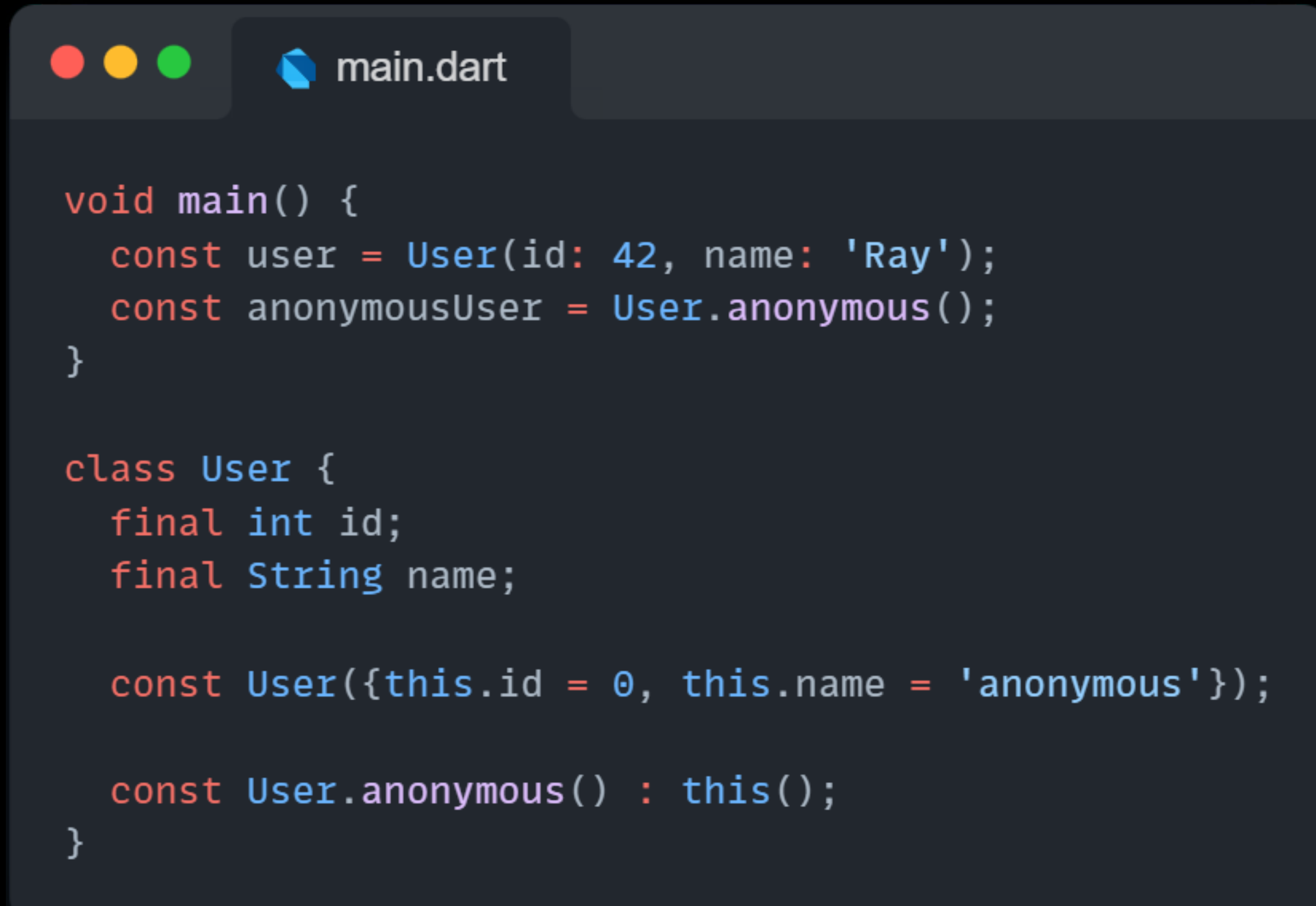
Exercise

- User class da parameterized constructor parametri optional named holatda va default qiymatlar sifatida anonymous constructor dagi qiymatlarni berib ozgartiring.
- Redirect constructor ni yangi constructorga moslang.
- Class field larini private qiling.

Constant constructors

- Biz class field larini tashqi o'zgarishlardan ximoyalashni allaqachon o'rgandik. Fieldlarni private qilish orqali.
- Bu muammoni boshqa usul bilan ham bartaraf etishimiz mumkun, class field larini immutable qilish orqali field larni keyingi o'zgarishlardan himoya qilishimiz mumkun. Endi fieldlarni private qilishimiz shart emas.
- Dart da o'zgaruvchilarni immutable qilishning ikki usuli mavjud: final va const
- Class field lari o'z qiymatini runtime da olganligi sababli fieldlarni faqat final qilishimiz mumkun.
- Agar ma'lum bir classning instance lari hech qachon o'zgarmasa, class ning barcha field lari final bo'lsa, class ning barcha instance lari compile-time constantlari bo'lishini tamillash uchun const keywordini constructor oldiga qo'yishingiz mumkun.

- User nomli class ni immutable qilamiz
- O'zgarmas bo'lishi bilan birga, agar class field lari ham bir xil bo'lsa dart bu instance larni bir deb biladi.



```
void main() {  
  const user = User(id: 42, name: 'Ray');  
  const anonymousUser = User.anonymous();  
}  
  
class User {  
  final int id;  
  final String name;  
  
  const User({this.id = 0, this.name = 'anonymous'});  
  
  const User.anonymous() : this();  
}
```

Factory constructors

- Siz hozirgacha ko'rgan barcha konstruktorlar generativ konstruktorlar hisoblanadi. Dart, shuningdek, zavod konstruktori deb ataladigan boshqa turdagi konstruktorni ham taqdim etadi.
- Factory constructor ob'ektlarni yaratishda ko'proq moslashuvchanlikni ta'minlaydi. Generativ konstruktor faqat class ning yangi nusxasini yaratishi mumkin. Biroq, zavod konstruktorlari class ning mavjud nusxalarini yoki hatto uning pastki class larini qaytarishi mumkin. Bu class ning amalga oshirish tafsilotlarini uni ishlatadigan koddan yashirishni xohlaganingizda foydalidir.
- Factory constructor asosan factory kalit so'zi bilan boshlanadigan va class tipidagi ob'ektni qaytaradigan maxsus method dir.

- Factory methodi user class ning yangi nusxasini yaratish va qaytarish uchun generativ konstruktordan foydalanadi.

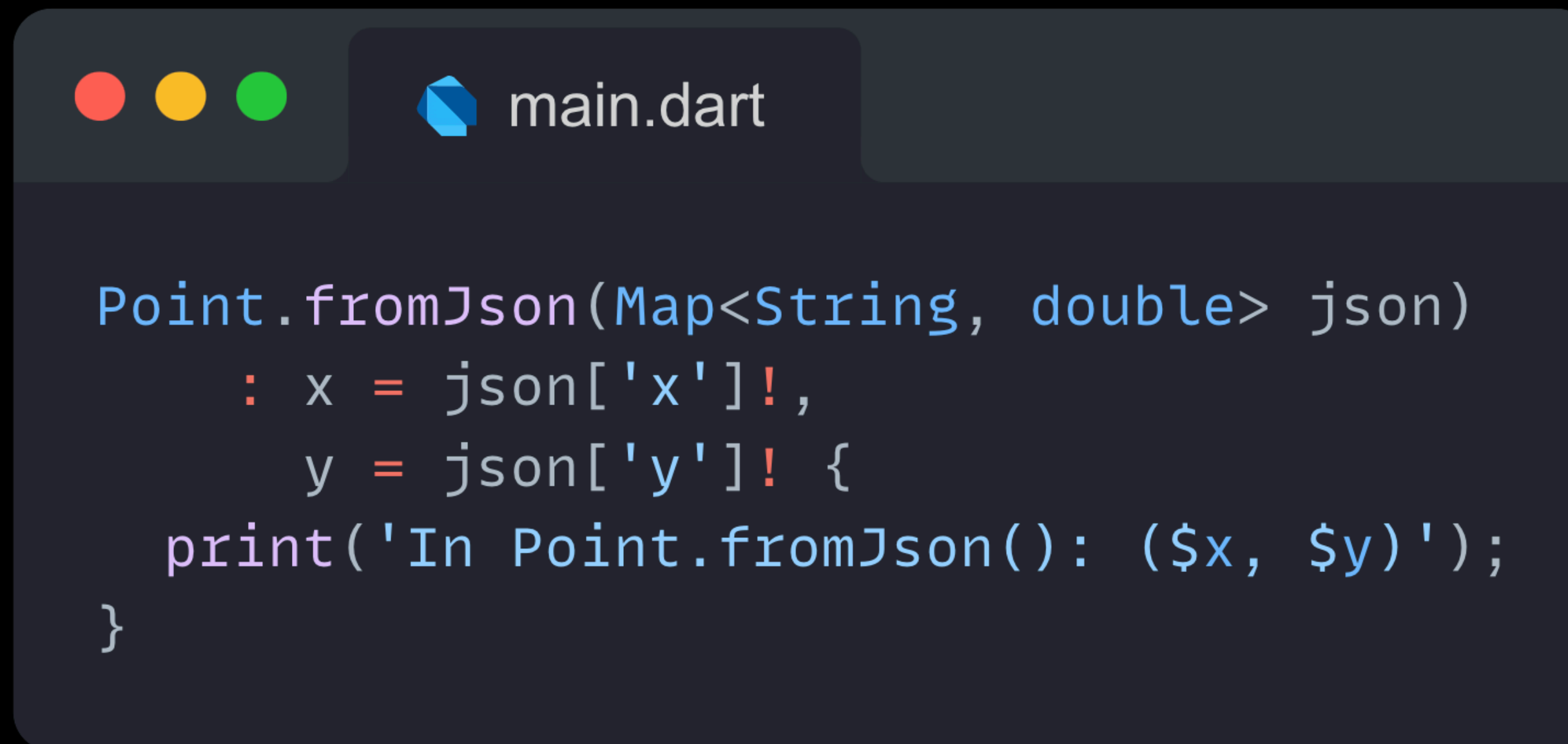


The image shows a code editor window with a dark theme. The title bar at the top has three colored circles (red, yellow, green) and a tab labeled 'main.dart' with a blue Dart logo icon. The code inside the editor is a Dart factory method for a 'User' class. The code is as follows:

```
factory User.ray() {  
    return User(id: 42, name: 'Ray');  
}
```

Initializer list

- Yuqori sinf konstruktorini chaqirishdan tashqari, konstruktor tanasi ishga tushgunga qadar instance o'zgaruvchilarini ishga tushirishingiz ham mumkin. Initsializatorlarni vergul bilan ajrating.
- Initializer ro'yxati konstruktor tanasi ishga tushishidan oldin instance o'zgaruvchilarini o'rnatadi.



```
Point.fromJson(Map<String, double> json)
  : x = json['x']!,
    y = json['y']! {
  print('In Point.fromJson(): ($x, $y)');
}
```

- Initializer ro'yxatlari final maydonlarni o'rnatishda qulaydir. Quyidagi misol boshlang'ich ro'yxatidagi uchta final maydonni ishga tushiradi.

```
import 'dart:math';

class Point {
  final double x;
  final double y;
  final double distanceFromOrigin;

  Point(double x, double y)
    : x = x,
      y = y,
      distanceFromOrigin = sqrt(x * x + y * y);
}

void main() {
  var p = Point(2, 3);
  print(p.distanceFromOrigin);
}
```

Exercise

- `final Map<String, dynamic> map = {'id': 10, 'name': 'Manda'};`
- `final manda = User.fromJson(map);`
- Yuqorida keltirilgan map ni qabul qiladigan va `User.fromJson()` named constructori orqali yangi instace yaratib qaytaradigan factory constructori ni yarating.

Summary

Home task: 1

1. Vehicle nomli klass tuzilsin. Bu klassni imkon qadar mukammal modellashtiring, yani fieldlariga yaxshi e'tibor berilishi kerak va shu klassda quyidagilar bo'lishi kerak:

- 1. generative parameterize constructor tuzilsin
- 2. truck nomli named constructor tuzilsin
- 3. bus nomli named constructor tuzilsin
- 4. sport nomli named constructor tuzilsin
- 5. car nomli named redirect constructor tuzilsin
- 6. toString methodi bo'lsin
- 7. == va hashCode override qilinsin
- 8. compareTo methodi bo'lsin
- 9. balonlar soni va o'rindiqlar soni uchun getter/setter yozilsin
- 10. main funksiyani ichida yuqoridagi constructorlardan foydalanib bir nechta object lar yasang, so'ng shu objectlarni listni ichiga joylang, so'ng sortlansin.

Home task: 2

- 2. Employee nomli klass tuzilsin. Bu klassni imkon qadar mukammal modellashtiring, yani fieldlariga yaxshi e'tibor berilishi kerak va shu klassda quyidagilar bo'lishi kerak:
 - 1. generative parameterize constructor tuzilsin
 - 2. intern nomli named constructor tuzilsin
 - 3. const constructor tuzilsin
 - 4. factory constructor tuzilsin
 - 5. toString metodi bo'lsin
 - 6. == va hashCode override qilinsin
 - 7. compareTo metodi bo'lsin
 - 8. getter/setter yozilsin
 - 9. main funksiyani ichida yuqoridagi constructorlardan foydalanib bir nechta object lar yasang, so'ng shu objectlarni listni ichiga joylang, so'ng sortlansin.

Home task: 3

- 3. Product nomli klass tuzilsin. Bu klassni imkon qadar mukammal modellashtiring, yani fieldlariga yaxshi e'tibor berilishi kerak va shu klassda quyidagilar bo'lishi kerak:
 - 1. generative parameterize constructor tuzilsin
 - 2. fruit nomli named constructor tuzilsin
 - 3. drink nomli named constructor tuzilsin
 - 4. const constructor tuzilsin
 - 5. factory constructor tuzilsin
 - 6. toString metodi bo'lsin
 - 7. == va hashCode override qilinsin
 - 8. compareTo metodi bo'lsin
 - 9. getter/setter yozilsin
 - 10. main funksiyani ichida yuqoridagi constructorlardan foydalanib bir nechta object lar yasang, so'ng shu objectlarni listni ichiga joylang, so'ng sortlansin.

Home task: 4

- 4. Computer nomli klass tuzilsin. Bu klassni imkon qadar mukammal modellashtiring, yani fieldlariga yaxshi e'tibor berilishi kerak va shu klassda quyidagilar bo'lishi kerak:
 - 1. generative parameterize constructor tuzilsin
 - 2. laptop nomli named constructor tuzilsin
 - 3. desktop nomli named constructor tuzilsin
 - 4. const constructor tuzilsin
 - 5. factory constructor tuzilsin
 - 6. toString methodi bo'lsin
 - 7. == va hashCode override qilinsin
 - 8. compareTo methodi bo'lsin
 - 9. getter/setter yozilsin
 - 10. main funksiyani ichida yuqoridagi constructorlardan foydalanib bir nechta object lar yasang, so'ng shu objectlarni listni ichiga joylang, so'ng sortlansin.

Q&A

Thank you for your time!