BELAJAR KOTLIN

1. Data

Bagus! Anda telah mempelajari variabel di Kotlin, alasan pentingnya variabel dalam pemrograman, serta cara membuat, memperbarui, dan menggunakannya. Anda telah bereksperimen dengan berbagai jenis data dasar di Kotlin, seperti jenis data Int, Double, String, dan Boolean. Anda juga telah mempelajari perbedaan antara kata kunci val dan var.

Semua konsep ini adalah elemen penting dalam perjalanan Anda untuk menjadi seorang developer.

Sampai jumpa di codelab berikutnya.

Ringkasan

* Variabel adalah penampung untuk satu bagian data.
* Anda harus mendeklarasikan variabel terlebih dahulu sebelum menggunakannya.
* Gunakan kata kunci val untuk menentukan variabel yang bersifat hanya baca jika tempat nilai tidak dapat berubah setelah ditetapkan.
* Gunakan kata kunci var untuk menentukan variabel yang dapat diubah atau diubah.
* Di Kotlin, sebaiknya gunakan val daripada var jika memungkinkan.
* Untuk mendeklarasikan variabel, mulai dengan kata kunci val atau var. Kemudian, tentukan nama variabel, jenis data, dan nilai awal. Misalnya: val count: Int = 2.
* Dengan inferensi jenis, hilangkan jenis data dalam deklarasi variabel jika nilai awal diberikan.
* Beberapa jenis data Kotlin dasar yang umum mencakup: Int, String, Boolean, Float, dan Double.
* Gunakan operator penetapan (=) untuk menetapkan nilai ke variabel selama deklarasi variabel atau memperbarui variabel.
* Anda hanya dapat memperbarui variabel yang telah dideklarasikan sebagai variabel yang dapat diubah (dengan var).
* Gunakan operator penambahan (++) atau operator pengurangan (--) untuk menambah atau mengurangi nilai variabel bilangan bulat masing-masing sebesar 1.
* Gunakan simbol + untuk menyambungkan string menjadi satu. Anda juga dapat menyambungkan variabel dari jenis data lain seperti Int dan Boolean ke Strings.

1. Function on KOTLIN

Selamat! Anda telah mempelajari cara menentukan dan memanggil fungsi di Kotlin.

Ringkasan

* Fungsi ditetapkan dengan kata kunci fun dan berisi bagian kode yang dapat digunakan kembali.
* Fungsi membantu membuat program yang lebih besar menjadi lebih mudah dikelola dan mencegah pengulangan kode yang tidak perlu.
* Fungsi dapat menampilkan nilai yang dapat Anda simpan dalam variabel untuk digunakan di lain waktu.
* Fungsi dapat menggunakan parameter, yaitu variabel yang tersedia di dalam isi fungsi.
* Argumen adalah nilai yang Anda teruskan saat memanggil fungsi.
* Anda dapat memberi nama argumen saat memanggil fungsi. Saat menggunakan argumen yang dinamai, Anda dapat menyusun ulang argumen tanpa memengaruhi output.
* Anda dapat menentukan argumen default yang memungkinkan Anda menghilangkan argumen saat memanggil fungsi.

1. Latihan

Mencetak pesan

Solusi ini menggunakan fungsi println() untuk mencetak pesan di setiap baris.

fun main() {  
    println("Use the val keyword when the value doesn't change.")  
    println("Use the var keyword when the value can change.")  
    println("When you define a function, you define the parameters that can be passed to it.")  
    println("When you call a function, you pass arguments for the parameters.")  
}

Memperbaiki error kompilasi

Kode ini berisi dua error kompilasi:

String harus diakhiri dengan tanda kutip ganda, bukan tanda kutip tunggal.

Argumen fungsi harus diakhiri dengan tanda kurung, bukan tanda kurung kurawal.

fun main() {  
    println("New chat message from a friend")  
}

Template string

Error kompilasi adalah hasil dari penetapan variabel hanya baca discountPercentage dan offer ke nilai baru; penetapan ini tidak diizinkan.

fun main() {  
    val discountPercentage = 20  
    val item = "Google Chromecast"  
    val offer = "Sale  - Up to $discountPercentage% discount off $item! Hurry Up!"  
  
    println(offer)  
}

Sebagai solusi alternatif, Anda dapat mendeklarasikan bilangan bulat discountPercentage dan string offer dengan kata kunci var. Namun demikian, nilainya tidak dapat diubah dalam konteks program sehingga Anda dapat tetap menggunakan kata kunci val.

Penyambungan string

Langkah 1

Program ini mencetak output ini:

The total party size is: 2030

Ini pertanyaan yang sulit. Ketika operator + digunakan pada nilai String, operator tersebut akan menghasilkan string gabungan. Bilangan bulat digabungkan dengan tanda kutip ganda sehingga diperlakukan sebagai string, bukan bilangan bulat, oleh dan menjadi output 2030.

Langkah 2

Anda dapat menghapus tanda kutip ganda di sekitar variabel numberOfAdults dan numberOfKids untuk mengonversinya ke variabel Int.

fun main() {  
    val numberOfAdults = 20  
    val numberOfKids = 30  
    val total = numberOfAdults + numberOfKids  
    println("The total party size is: $total")  
}

Jika Anda ingat, compiler Kotlin dapat menyimpulkan jenis variabel berdasarkan nilai yang telah ditetapkan padanya. Dalam hal ini, compiler akan menyimpulkan bahwa variabel numberOfAdults dan numberOfKids adalah jenis Int.

Pemformatan pesan

Program ini mencetak output ini:

Congratulations for your bonus! You will receive a total of 5000 + 1000 (additional bonus).

"$baseSalary + $bonusAmount" menggunakan sintaksis ekspresi template. Dalam ekspresi template, kode dievaluasi terlebih dahulu, lalu hasilnya digabungkan dalam string.

Dalam pertanyaan, variabel $baseSalary dievaluasi ke nilai 5000 dan variabel $bonusAmount dievaluasi ke nilai 1000. Kemudian, hasilnya digabungkan untuk menghasilkan "5000 + 1000" dan ditetapkan ke variabel result.

Mengimplementasikan operasi matematika dasar

Langkah 1

Tentukan variabel result yang tidak dapat diubah dengan kata kunci val, lalu tetapkan hasil dari operasi penambahan:

fun main() {  
    val firstNumber = 10  
    val secondNumber = 5  
  
    val result = firstNumber + secondNumber  
    println("$firstNumber + $secondNumber = $result")  
}

Langkah 2

Buat fungsi add() yang menerima parameter firstNumber dan parameter secondNumber, keduanya dari jenis Int, dan menampilkan nilai Int.

Masukkan kode untuk operasi penambahan di dalam isi fungsi add(), lalu gunakan kata kunci return untuk menampilkan hasil operasi.

fun add(firstNumber: Int, secondNumber: Int): Int {  
    return firstNumber + secondNumber  
}

Langkah 3

Tentukan fungsi subtract() yang menerima parameter firstNumber dan parameter secondNumber, keduanya dari jenis Int, dan menampilkan nilai Int.

Masukkan kode untuk operasi pengurangan di dalam isi fungsi subtract(), lalu gunakan kata kunci return untuk menampilkan hasil operasi.

fun subtract(firstNumber: Int, secondNumber: Int): Int {  
    return firstNumber - secondNumber  
}

Ubah fungsi main() untuk menggunakan fungsi subtract() yang baru. Contoh solusi dapat terlihat seperti ini:

fun main() {  
    val firstNumber = 10  
    val secondNumber = 5  
    val thirdNumber = 8  
  
    val result = add(firstNumber, secondNumber)  
    val anotherResult = subtract(firstNumber, thirdNumber)  
  
    println("$firstNumber + $secondNumber = $result")  
    println("$firstNumber - $thirdNumber = $anotherResult")  
}  
  
fun add(firstNumber: Int, secondNumber: Int): Int {  
    return firstNumber + secondNumber  
}  
  
fun subtract(firstNumber: Int, secondNumber: Int): Int {  
    return firstNumber - secondNumber  
}

Parameter default

Langkah 1

Buat fungsi displayAlertMessage() yang menerima parameter operatingSystem dan parameter emailId, keduanya dari jenis String, dan menampilkan nilai String.

Dalam isi fungsi, gunakan ekspresi template untuk memperbarui pesan dan menampilkannya.

fun displayAlertMessage(operatingSystem: String, emailId: String): String {  
    return "There is a new sign-in request on $operatingSystem for your Google Account $emailId."  
}

Langkah 2

Tetapkan nilai "Unknown OS" ke parameter operatingSystem.

fun displayAlertMessage(  
    operatingSystem: String = "Unknown OS",  
    emailId: String  
): String {  
    return "There is a new sign-in request on $operatingSystem for your Google Account $emailId."  
}

Pedometer

Nama fungsi dan nama variabel harus mengikuti konvensi camel case.

Jika nama berisi beberapa kata, huruf kecil di huruf pertama dari kata pertama, menggunakan huruf besar pada huruf pertama dari kata berikutnya, dan menghapus spasi di antara kata tersebut.

Contoh nama fungsi meliputi:

calculateTip

displayMessage

takePhoto

Contoh nama variabel meliputi:

numberOfEmails

cityName

bookPublicationDate

Untuk mempelajari nama lebih lanjut, lihat [Aturan penamaan](https://kotlinlang.org/docs/coding-conventions.html#naming-rules).

Hindari penggunaan [kata kunci Kotlin](https://kotlinlang.org/docs/keyword-reference.html) sebagai nama fungsi karena kata tersebut sudah memiliki arti tertentu dalam bahasa Kotlin.

Kode solusi Anda akan terlihat seperti cuplikan kode ini:

fun main() {  
    val steps = 4000  
    val caloriesBurned = pedometerStepsToCalories(steps);  
    println("Walking $steps steps burns $caloriesBurned calories")  
}  
  
fun pedometerStepsToCalories(numberOfSteps: Int): Double {  
    val caloriesBurnedForEachStep = 0.04  
    val totalCaloriesBurned = numberOfSteps \* caloriesBurnedForEachStep  
    return totalCaloriesBurned  
}

Membandingkan dua angka

Buat fungsi compareTime() yang menerima parameter timeSpentToday dan parameter timeSpentYesterday, keduanya dari jenis Int, dan menampilkan nilai Boolean.

Solusi ini bergantung pada operator perbandingan >. Operator akan bernilai Boolean sehingga fungsi compareTime() hanya menampilkan hasil timeSpentToday > timeSpentYesterday.

Misalnya, jika Anda meneruskan argumen 300 ke parameter timeSpentToday dan argumen 250 ke parameter timeSpentYesterday, isi fungsi akan bernilai 300 > 250 yang menampilkan nilai true, karena 300 lebih besar dari 250.

fun main() {  
    println("Have I spent more time using my phone today: ${compareTime(300, 250)}")  
    println("Have I spent more time using my phone today: ${compareTime(300, 300)}")  
    println("Have I spent more time using my phone today: ${compareTime(200, 220)}")  
}  
  
fun compareTime(timeSpentToday: Int, timeSpentYesterday: Int): Boolean {  
    return timeSpentToday > timeSpentYesterday  
}

Have I spent more time using my phone today: true

Have I spent more time using my phone today: false

Have I spent more time using my phone today: false

Memindahkan kode duplikat ke fungsi

Buat fungsi yang akan mencetak detail cuaca untuk kota Jakarta setelah fungsi main().

Untuk nama fungsi, Anda dapat menggunakan printWeatherForCity() atau yang serupa.

Aktifkan fungsi dari fungsi main().

Program tersebut akan mencetak detail cuaca untuk Jakarta.

fun main() {  
    printWeatherForCity()  
}  
  
fun printWeatherForCity() {  
    println("City: Ankara")  
    println("Low temperature: 27, High temperature: 31")  
    println("Chance of rain: 82%")  
    println()  
}

Sekarang Anda dapat membuat fungsi lain yang lebih dinamis sehingga dapat mencetak detail cuaca untuk kota lain.

Ganti bagian khusus Jakarta dari pernyataan println() dengan variabel.

Jangan lupa gunakan konvensi camel case untuk nama variabel dan simbol $ sebelum variabel sehingga nilai variabel tersebut yang digunakan, bukan nama variabel. Ini adalah template string yang telah Anda pelajari di codelab sebelumnya.

fun printWeatherForCity() {  
    println("City: $cityName")  
    println("Low temperature: $lowTemp, High temperature: $highTemp")  
    println("Chance of rain: $chanceOfRain%")  
    println()  
}

Ubah definisi fungsi sehingga variabel tersebut merupakan parameter yang harus diteruskan ke fungsi saat dipanggil, lalu tentukan jenis data untuk setiap parameter.

Parameter cityName adalah jenis String, sedangkan parameter lowTemp, highTemp, dan chanceOfRain adalah jenis Int.

Tidak ada nilai return yang diperlukan dalam definisi fungsi karena pesan dicetak ke output.

fun printWeatherForCity(cityName: String, lowTemp: Int, highTemp: Int, chanceOfRain: Int) {  
    println("City: $cityName")  
    println("Low temperature: $lowTemp, High temperature: $highTemp")  
    println("Chance of rain: $chanceOfRain%")  
    println()  
}

Perbarui fungsi main() untuk mengaktifkan fungsi printWeatherForCity() dan masukkan detail cuaca untuk Jakarta.

Nama kotanya "Ankara", suhunya rendahnya 27, suhu tertingginya 31, dan peluang turunnya hujan 82.

fun main() {  
    printWeatherForCity("Ankara", 27, 31, 82)  
}  
  
fun printWeatherForCity(cityName: String, lowTemp: Int, highTemp: Int, chanceOfRain: Int) {  
    println("City: $cityName")  
    println("Low temperature: $lowTemp, High temperature: $highTemp")  
    println("Chance of rain: $chanceOfRain%")  
    println()  
}

Jalankan program untuk memastikan bahwa output menampilkan detail cuaca untuk Jakarta.

Aktifkan fungsi printWeatherForCity() dengan detail cuaca untuk kota lain.

fun main() {  
    printWeatherForCity("Ankara", 27, 31, 82)  
    printWeatherForCity("Tokyo", 32, 36, 10)  
    printWeatherForCity("Cape Town", 59, 64, 2)  
    printWeatherForCity("Guatemala City", 50, 55, 7)  
}  
  
fun printWeatherForCity(cityName: String, lowTemp: Int, highTemp: Int, chanceOfRain: Int) {  
    println("City: $cityName")  
    println("Low temperature: $lowTemp, High temperature: $highTemp")  
    println("Chance of rain: $chanceOfRain%")  
    println()  
}

Jalankan program.

Tindakan ini akan mencetak output yang sama seperti program asli, tetapi sekarang kode Anda lebih ringkas dan tidak berisi pengulangan yang tidak perlu! Semua kode untuk mencetak detail cuaca kota dipusatkan di satu tempat: fungsi printWeatherForCity(). Jika ingin mengubah tampilan detail cuaca, Anda dapat mengubahnya di satu tempat yang berlaku untuk semua kota.