

**PROYECTO USAID PUENTES PARA EL EMPLEO**

**CURSO PROGRAMADOR ANALISTA JAVA**



**TRABAJO DE INVESTIGACION, KOTLIN**

**PRESENTADO POR:**

**AMANDA AYALA QUINTANILLA**

**ERICK UVALDO GOMEZ FLORES**

**FREDY ALEXI ROMERO PEREZ**

**NAHOMY SUGEY VELAZQUEZ MONTYES**

**MARIA TERESA AGUILLON ELIAS**

**RONALD BALMORE REYES SANCHEZ**

**DICIEMBRE, 2019**

**SAN SALVADOR, EL SALVADOR, CENTROAMERICA.**

**INDICE**

**Contenido Pag**

1.1 Objetivos………………………………………………………………….3

1.1.1 Objetivos Generales…………………………………………3

1.1.2 Objetivos Especificos………………………………………..3

1.2 Definicion de Kotlin……………………………………………………4

**1.1 OBJETIVOS.**

**1.1.1 General.**

* Investigar que es Kolin y sus herramientas.
* Aprender a como instalar Kotlin y sus complementos.
* Comprender el funcionamiento de las estucturas de control con Kotlin.
* Aprender sobre los arreglos en el Lenguaje de Programación Kotlin.
* Conocer los Frameworks de Kotlin
* Mostrar las herramientas disponibles para Kotlin.

**1.1.2 Específicos**

* Conceptuar que es Kotlin y lograr comprender su funcionamiento.
* Detallar con pasos a seguir para la instalación y su respectiva desinstalación.
* Deducir el funcionamiento de las estructuras de control en Kotlin.
* Descubrir como Kotlin maneja los arreglos.
* Listar los frameworks disponibles para Kotlin
* Identificar las herramientas para crear un entorno de dasarrollo para trabajar con Kotlin.

**1.2 Kotlin.**

Es un lenguaje de programación (a veces conocido como swift de Android) de tipado estático que corre sobre la máquina virtual de Java y que también puede ser compilado a código fuente de JavaScript.

Lo que lo hace útil en el desarrollo de Android es que compila a bytecode JVM, y también se puede compilar con JavaScript. Es totalmente compatible con Java y el código de Kotlin puede ser simplemente convertido a código Java y viceversa (hay un *plugin* de JetBrains). Esto significa que Kotlin puede usar cualquier marco, biblioteca, etc., escrito en Java. En Android, se integra por Gradle.

**Instalación de Kotlin (Windows).**

**Instalación de IDE IntelliJ IDEA para Kotlin.**

* Inicialmente debemos descargar e instalar el JDK.
* Luego descargamos el IntelliJ IDEA Community.
* Ejecutamos y dejamos los datos de instalación por defecto.
* Ejecutamos una vez terminada la instalación y nos preguntara si deseamos importar alguna configuración y seleccionamos que no y ok.
* Aceptamos los términos de uso y le damos en continuar.
* Luego aparece una ventana preguntando si queremos o no compartir datos, escogemos cualquiera de las dos opciones ya que solo es para estudios de estadísticas, rendimiento de hardware y software y sobre todo no se comparten datos sensibles (personales y código fuente).
* Nos permitirá seleccionar el color de entorno blanco u oscuro y seleccionamos el que nos guste y le damos en el botón skip remaining and set default.
* Luego nos aparecerá la pantalla para iniciar, abrir o importar proyecto y listo.

**Instalando plugin de kotlin para IDE eclipse.**

* Abrimos el eclipse.
* Luego en el menú de eclipse nos vamos a HelpàMarketplace.
* Buscamos kotlin.
* Encontraremos Kotlin Plugin for Eclipse y le damos en install.
* Aceptamos los términos de uso y le damos en el botón finish.
* Al terminar la instalación del plugin pedirá reiniciar eclipse y le damos en aceptar.
* Al reiniciar eclipse buscamos en el menú la opción WindowsàPerspectiveàOpen PerspectiveàOther.
* Seleccionamos Kotlin le damos en el botón ok.

y con esto tenemos todo listo para utilizar kotlin.

**Estructuras de control en Kotlin**

Kotlin tiene su propio uso de estructura de control, pero mantiene el soporte de las estructuras de control de JAVA.

**Expresion if:**

La estructura if se puede usar como una expresión o como una declaración.

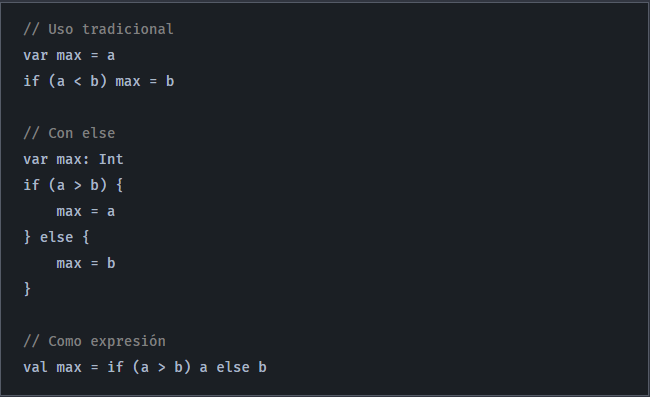


Figura 1. Estructura if.

**Expresion When:**

La estructura when es el sustituto de la estructura switch.

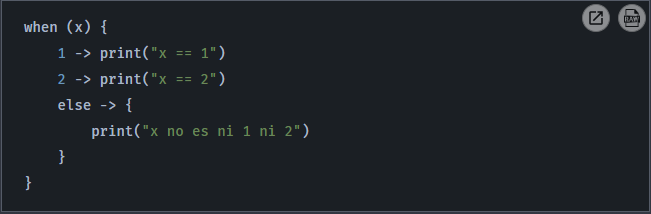
****

Figura 2. Estructura when.

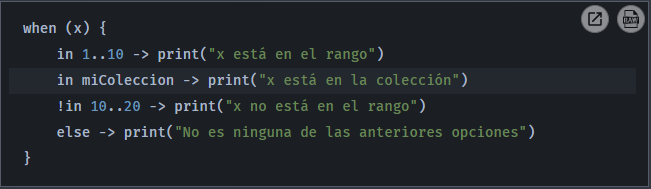
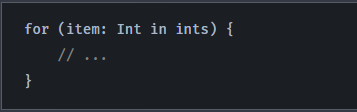


Figura 3. Estructura when con rangos.

**Bucles For:**

La estructura se puede declarar de varias maneras.

****

Figura 4. Declaracion de estructura for.

*Figura 5. Declaracion de estructura for en bloque.*

Para iterar a través de un rango de números, utiliza una expresión de rango:

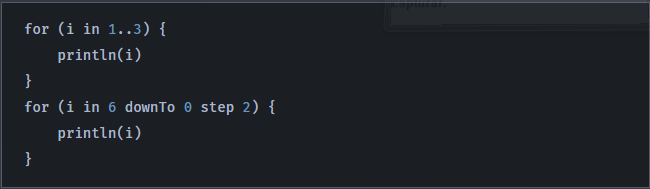
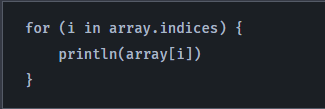


Figura 6. Declaracion de for con rango.

Para iterar a través de un array o una lista con un índice se haría del siguiente modo:



*Figura 7. Declaracion de for iterando arreglo.*

**Bucles While:**

El bucle while y do while funcionan como en lenguaje java no presenta mayo cambio.

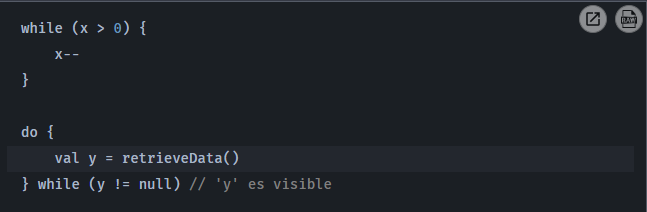
**

Figura 8. Estructura while y do-while.

**Arreglos en Kotlin.**

Las arrays (o arreglos) son secuencias de datos, del mismo tipo e identificados por un nombre común. Para hacerlo más fácil de entender imaginemos que tenemos que almacenar los 7 días de la semana, podríamos crear 7 variables Strings o almacenarlas todas en un solo array.



Tabla 1. Creando un arreglo.

Ahora la variable **weekDays**contiene todos los días de la semana. Ahora para acceder a cada uno de los valores, lo haremos a través de la posición. Por ejemplo, imaginemos un edificio, cada valor se almacena en una planta, por lo que el primer valor estará en la posición 0, el segundo en la 1 y así con cada uno de ellos. Hay que recordar que se empieza en la posición 0.

Entonces podremos acceder a cada uno de los valores gracias a la función *get()*que nos devolverá el valor de dicha posición.

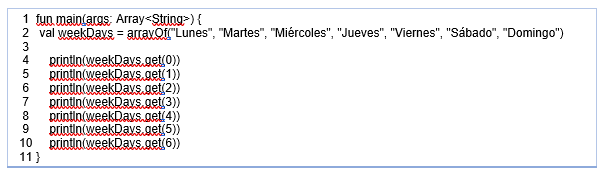


Tabla 2. Imprimiendo las posiciones del arreglo.

Esto nos pintará los días de la semana (posiciones de la 0 a la 6). Si pusiéramos en el get una posición que no tiene, por ejemplo, la 7 nos daría una excepción al ejecutarse la aplicación *ArrayIndexOutOfBoundsException,*y es por ello por lo que al trabajar con arrays debemos tener bien claro el tamaño del array.

Para evitar ese tipo de problemas podemos usar la función size, que nos devolverá el tamaño de dicho array.

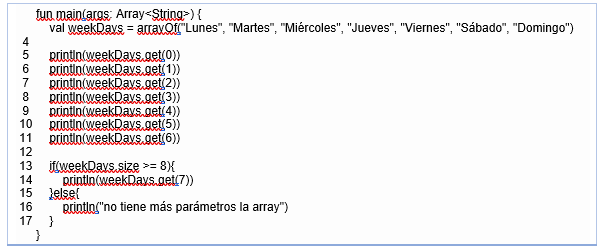


Tabla 3. Buscando el tamaño del arreglo con la función size.

Si ejecutamos dicho código no entrará en el if porque el tamaño del array es de 7 (No es lo mismo tamaño que posición), por lo que nos pintará «no tiene más parámetros en el array».

Como había dicho al principio del post, las array tienen una serie de limitaciones, entre ellas que tienen que tener un tamaño fijo y será el número de valores que le asignemos al instanciarla, eso significa que siempre va a tener un tamaño de 7, y no podremos añadir más datos, pero si cambiarlos a través de la función *set()*.



Tabla 4. Cambiando valores del arreglo.

La función *set()* recibe dos parámetros, el primero es la posición a la que queremos acceder y el segundo el es nuevo valor a reemplazar. Hay que tener en cuenta que el valor que le mandemos debe ser del mismo tipo, por ejemplo, este array son de Strings, por lo que no podemos pasar un Int.

**Recorriendo Arrays**

Ahora que ya conocemos un poco más que es un array y como trabajar con ellas, vamos a ver como recorrerlas.

Aunque podríamos hacerlo como hicimos arriba, hay modos más rápidos y óptimos, el bucle *for()*. Este bucle nos permite entre otras, recorrer el array entero, posición por posición y acceder a cada uno de los parámetros que contiene. Vamos a volver a pintar los 7 días de la semana con este método.

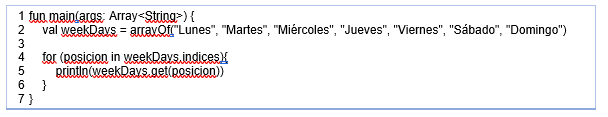


Tabla 5. Recorriendo al arreglo.

Mucho menos código ¿verdad? Vamos a ver qué es lo que hemos hecho.

El for necesitará una variable, en este caso «posición» que irá teniendo el valor de cada una de las posiciones del array. Su funcionamiento es muy sencillo, cuando pasa por el for por primera vez, tendrá valor 0, entonces comprueba el tamaño de *weekDays* y si es mayor, entra a la función, hace lo que le pidamos (en este caso pintar en la posición de la variable) y vuelve al inicio, así hasta llegar a 6 que será la última posición del array.

También el for nos permite sacar tanto el índice como el valor directamente, para ello haríamos lo siguiente.

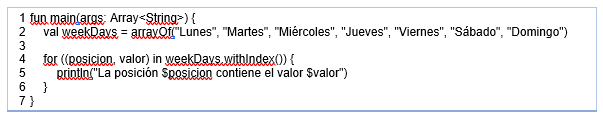


Tabla 6. Utilizando la función For para que nos imprima cada una de las posiciones.

Y dicho for nos devolvería lo siguiente

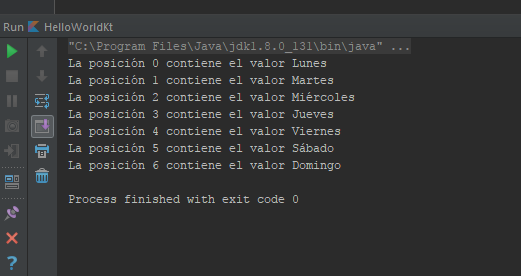


Figura 9. Formas y descripción de las formas.

Si por el contrario solo os interesa el contenido podríamos hacer directamente un *for in*sin acceder a la posición, solo al contenido.



Tabla 7. Devolviendo la lista completa del arreglo.

Devolviéndonos la lista de días de la semana.

**Framework´s.**

Kotlin abrió un mundo nuevo para el mundo del desarrollo de aplicaciones. Desde su aparición, y en especial desde la versión 1.1, los desarrolladores escriben mucho menos código en sus proyectos gracias a su interoperabilidad con Java, escalabilidad y su sintaxis simplificada que hace más fácil incorporarlo a aplicaciones nuevas o existentes, al tiempo que podemos usar nuestras librerías favoritas para Java.

Otro punto importante a su favor es la incorporación de dos grandes novedades como son las typealiases y las co-rutinas. Las primeras tienen múltiples aplicaciones diferentes; pero, la más interesante reside en la capacidad de hacer nuestros listeners más legibles al cambiar el nombre de los tipos complejos a otros más legibles.

Las co-rutinas (coroutines) son métodos que tienen la capacidad de pausar y reiniciar exactamente donde se quedó en el frame anterior. En otras palabras, las co-rutinas permiten escribir código asíncrono de manera síncrona, lo que permite suspender su ejecución en algún momento y esperar a un resultado. Quizá el concepto genere un poco de confusión al comienzo porque no son una biblioteca o una implementación específica, son una característica del lenguaje que permite crear las bibliotecas sobre ella.

La potencia que Kotlin trae a los desarrolladores está fuera de toda duda, incluso podría decirse que es ya un lenguaje oficial en Android.

Los diferentes entornos de trabajo disponibles compatibles con Kotlin nos permiten todo eso y más sin que tengamos que reinventar la rueda en complejas inyecciones de dependencias.

A continuación, se muestran algunos de los frameworks de Kotlin.

**Framework Spring.**

Uno de los framework más veteranos y uno de nuestros favoritos. En su momento nació debido a las limitaciones de Java EE y la verbosidad con la que se escribían cualquier tipo de aplicaciones Java.

Con el tiempo ha evolucionado y se ha posicionado por delante de sus competidores en cuanto hablamos de desarrollo web, cloud, backend o microservicios. Framework Spring es ahora un estándar de facto entre desarrolladores.

Desde su versión 5.0 ofrece soporte a Kotlin destacando por su interoperabilidad, es decir, no es necesario para usar Kotlin crear un proyecto desde cero, todas las clases Java se pueden usar en Kotlin y viceversa.

Este framework puede descargarse desde los sitios repo.spring.io o Maven Central.

Características:

* Tecnologías, Spring es completamente modular y soporta diferentes tecnologías como la inyección de dependencias, eventos, recursos.
* Acceso a datos, soporte DAO, JDBC, ORM, Marshalling XML.
* Gestión de transacciones.
* Integración.
* Pruebas (Testing).
* Programación orientada a aspectos (AOP).
* Facilita en gran medida la programación basada en MVC (Modelo Vista Controlador).
* Es un Framework que tiene un especial foco sobre la Seguridad.

**Framework Spark:** es un conjunto de librerías para el desarrollo de aplicaciones web en Java inspirado en el framework Sinatra para Ruby, incluye por defecto el servidor de aplicaciones Jetty de manera que las aplicaciones web creadas con Spark pueden lanzarse como cualquier programa Java. Adicionalmente, las aplicaciones pueden configurarse para ser instaladas bajo otros servidores como por ejemplo Apache Tomcat.

Características:

* Está integrado con Apache Hadoop.
* Trabaja en memoria.
* Permite trabajar en disco.
* Proporciona API para Java, Scala, Python y R.
* Permite el procesamiento en tiempo real.
* Resilient Distributed Dataset (RDD): Usa la evaluación perezosa, lo que significa es que todas las transformaciones que vamos realizando sobre los RDD, no se resuelven, sino que se van almacenando en un grafo acíclico dirigido (DAG), y cuando ejecutamos una acción, es decir, cuando la herramienta no tenga más opción que ejecutar todas las transformaciones, será cuando se ejecute.

**Framework Ktor:** es un marco web asincrónico escrito y diseñado para Kotlin. Permitiendo que las características más impresionantes de Kotlin, como las corutinas, no solo se usen, sino que también sean compatibles.

**Herramientas de Desarrollo.**

**IntelliJ IDEA:** Es un entorno de desarrollo integrado que es desarrollado por JetBrains, también es uno de los editores más rápidos de Java.

Algunas características son:

* Soporte para plugins.
* Multilenguaje y multiplataforma.
* Soporte para herramientas como maven y sistemas de test con cobertura.
* Herramientas de bases de datos.
* Soporte de Frameworks.
* Soporte de servidores.

**JetBrains:** Es una compañía de desarrollo de software cuyas herramientas están dirigidas a desarrolladores de software y gerentes de proyectos.

**Apache Maven:** Es una herramienta que estandariza la configuración de un proyecto en todo su ciclo de vida, también la integración continua para poder realizar la ejecución de test unitarios y pruebas automatizadas.

Principales características:

* Sistema de gestión dependencias.
* Mecanismo de distribución de librerías.
* Mecanismo para ser extensible, por la creación de plugins customizables.
* Multi-plataforma, funciona con entorno Linux y Windows al ser una aplicación Java.
* Software libre.
* Fomenta la reutilización de código y librerías.
* Compatible con múltiples IDEs.

**Apache Ant:** Herramienta realizada para tareas mecánicas y repetitivas. Es un software para el proceso de automatización de compilación.

Características:

* Es la herramienta más utilizada para la construcción de proyectos.
* Integra pruebas JUnit con el proceso de compilación facilitando el desarrollo basado en pruebas.

**Gradre:** Sistema de automatización de código abierto que construye sobre los conceptos de Apache Ant y Apache Maven e introduce un lenguaje específico del dominio (DSL) basado en Groovy en vez de la forma XML utilizada por Apache Maven para declarar la configuración de proyecto.

Características:

* Depuración colaborativa.
* Construcción incremental.
* Diseño de repositorio personalizado.
* Dependencias transitivas.
* Soporte a Groovy y Scala incorporado.
* Compilación incremental para Java.
* Embalaje y distribución de JAR, WAR y EA.
* Integración con Android Studio.

**Android Studio:** Es el entorno de desarrollo integrado oficial para la plataforma Android. Está basado en el software IntelliJ IDEA de JetBrains.

Características:

* Integración de ProGuard y funciones de firma de aplicaciones.
* Más especificación a la hora de Programar.
* -Renderizado en tiempo real.
* -Soporte para construcción basada en Gradle.
* -Refactorización específica de Android y arreglos rápidos.
* -Un editor de diseño enriquecido que permite a los usuarios arrastrar y soltar componentes de la interfaz de usuario.

**NetBeans:** es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java, proyecto de código abierto.

Características:

* Proporciona una base modular y extensible para el desarrollo de aplicaciones.
* Incluye servicios para el control de interfaz de usuario, la configuración, el almacenamiento.
* Permite el desarrollo en otros lenguajes, mediante paquetes adicionales.

**Conclusiones.**