



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 1

CONTENIDO

1.	Disposiciones generales para cada proyecto:	3
	Documentación de desarrollos de software	3
	Especificaciones para el desarrollo de proyectos	4
2.	Implementación del estándar	6
	EQUIPO DE TRABAJO	6
	BASES DE DATOS	6
	PROYECTO	7
	ESTRUCTURA DE CARPETAS Y SUBCARPETAS DEL PROYECTO EN DESARROLLO	9
	ESTRUCTURA DE CÓDIGO	10
	TABLAS DE REFERENCIA	14

	Estándar de codificación para desarrollos de software	Código: ECDS-E01
		Versión: 1.0.0
		Fecha: 04-10-2019
		Página: 2

LISTADO DE FIGURAS

Figura 1. Guía codificación de identificador de proyecto	8
Figura 2. Sistema de versiones para desarrollos de software.....	8

LISTADO DE TABLAS

Tabla 1. Tipos de desarrollo	4
Tabla 2. Lenguajes de programación	5
Tabla 3. Entornos de desarrollo	5
Tabla 4. Sistemas operativos.....	5
Tabla 5. Manejadores de bases de datos.....	5
Tabla 6. Seguridad en los desarrollos de software	6
Tabla 7. Ejemplo asignación de código equipo de trabajo.....	6
Tabla 8. Ejemplo asignación de códigos de roles en bases de datos.	7
Tabla 9. Ejemplo asignación de códigos de tablas en bases de datos.	7
Tabla 10. Asignación de código según el nombre del proyecto.....	8
Tabla 11. Codificación para requerimientos de proyectos.	8
Tabla 12. Ejemplo sistema de versiones para desarrollos de software	9

LISTADO DE TABLAS DE REFERENCIA

Tabla R 1. Códigos para áreas del centro de desarrollo	14
<i>Tabla R 2. Códigos para unidades del centro de desarrollo</i>	<i>14</i>
Tabla R 3. Códigos para áreas del centro de desarrollo	15
Tabla R 4. Códigos para áreas del centro de desarrollo.....	15

	Estándar de codificación para desarrollos de software	Código: ECDS-E01
		Versión: 1.0.0
		Fecha: 04-10-2019
		Página: 3

ESTANDAR DE CODIFICACIÓN PARA DESARROLLOS DE SOFTWARE

La presente guía contiene los estándares de codificación para el desarrollo y documentación de proyectos y recursos software que sean asignados al área de desarrollo de la empresa Interlink S.A.

1. Disposiciones generales para cada proyecto:

- Cada proyecto se debe encontrar almacenado en la plataforma asignada al formato de inventario de proyectos.
- Una vez el estado del proyecto en el formato de inventario se encuentre en **aprobado** se debe iniciar a aplicar el estándar de codificación descrito. En este contexto el desarrollo de software se debe realizar bajo el esquema de los estados establecidos en el formato de inventarios: solicitado, aprobado, asignado, en desarrollo, completado, diferido, finalizado y cancelado.
- Cada uno de los proyectos debe contar con un modelo de base de datos relacional o no relacional que satisfaga los requerimientos establecidos en el formato de inventario.
- **Documentación de desarrollos de software:**
 - Cada uno de los proyectos debe contar con su correspondiente documentación en cada uno de los estados establecidos en el formato de acuerdo a los sistemas de gestión establecidos anteriormente. (el contenido que se dispone en este apartado se debe incluir en la documentación como lo orienta la [guía de documentación de información](#)).
 - **Solicitud de desarrollo:** debe incluir

Justificación: Contextualización, objeto, categorización y propuesta.

Requerimientos: especificaciones de requerimientos funcionales y no funcionales.

Listado de riesgos: posibles riesgos que podrían impactar el proceso de desarrollo.

Aplicabilidad: Sectores a los cuales va dirigido.



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 4

Metodología y despliegue: actividades necesarias para dar cumplimiento a los requerimientos establecidos y esquema de infraestructura o modelo funcional del desarrollo.

- **Aprobación y asignación**

Se estudia la solicitud de desarrollo, se aprueba por parte del director de área y se asigna la persona responsable y su equipo de trabajo.

- **Desarrollo**

Plan de proyecto: Cronograma de desarrollo de acuerdo a la metodología planteada.

Formato de productos en explotación: diligenciamiento del formato en donde se resumen y agrupan los componentes del sistema necesarios para la fase de pruebas.

Pruebas, detección y corrección de errores: lanzamiento de pruebas Alpha y Beta en las diferentes plataformas para detección y corrección de errores.

- **Transición**

Se especifica si el desarrollo fue completado es decir se da el visto bueno de su funcionalidad de la mano de los requerimientos y se lanza a producción de las diferentes plataformas, o por el contrario diferido o cancelado.

- **Especificaciones para el desarrollo de proyectos:**

- **Tipos de desarrollo**

Tipo	Código
Escritorio	ES
Web	WB
Móvil	MV

Tabla 1. Tipos de desarrollo

- **Lenguajes de programación que se manejan**

Lenguaje	Código
PHP	PHP



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 5

Dart	DR
Swift	SW
TypeScript	TS
JavaScript	JS
Java	JV
Kotlin	KT
Interprete Shell	SH
Python	PY

Tabla 2. Lenguajes de programación

○ Entornos de desarrollo

Nombre	Tipo	Código
Visual Estudio Code	Web/móvil	VSC
Android Studio	Móvil	WB
Xcode	Móvil	XC
Sublime Text	Web / móvil	ST

Tabla 3. Entornos de desarrollo

○ Sistemas operativos

Nombre	Tipo	Código
Windows	Web	WD
Mac OS	Web	MO
Android	Móvil	AD
IOS	Móvil	IOS

Tabla 4. Sistemas operativos

○ Manejadores de bases de datos

Manejador	Código
PostgreSQL	PSQL

Tabla 5. Manejadores de bases de datos

○ Seguridad en los desarrollos de software



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 6

Objeto	Web	Móvil
Seguridad de la información	1. Autenticación base de datos 2. Certificado digital SSL. 3. Cifrado MD5	1. Autenticación base de datos. 2. Cifrado MD5
Control de versiones	4. Uso de GIT y GITHUB (privado)	3. Uso de GIT y GITHUB (privado)
Respallos	5. VPS	4. VPS

Tabla 6. Seguridad en los desarrollos de software

2. Implementación del estándar:

EQUIPO DE TRABAJO

Cada miembro del equipo de trabajo del área de desarrollo de la empresa será identificado mediante un código asignado de forma individual, teniendo como base el modelo estructural del área. Lo anterior, con el fin de llevar un registro de control de las intervenciones en el proyecto de cada miembro tanto en documentación como en la escritura de código de desarrollo. Los códigos asignados para áreas y unidades se encuentran en la [Tabla R1](#) y [Tabla R2](#).

Área	Unidad	Nombre	Código numérico	Código asignado
Desarrollo y nuevas tecnologías	Programación	Yeins Rosio Garcia Bautista	97	DT_P_Y97

Tabla 7. Ejemplo asignación de código equipo de trabajo

BASES DE DATOS

Para el modelado de bases de datos se deben tener en cuenta las siguientes indicaciones:



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 7

Código de Roles: Se establecen roles para la gestión de privilegios de acceso de los grupos de usuarios. El identificador de cada rol asociado a cada proyecto se asigna de la siguiente forma (los códigos de la columna de roles se encuentran en la [Tabla R3](#)):

Tipo	Identificador de aplicación	Rol	Código asignando
R	PL-APH	A	R_PL-APH_A

Tabla 8. Ejemplo asignación de códigos de roles en bases de datos.

Tablas: Las tablas deben tener nombres que describan su contenido, el nombre debe ser singular y el código debe incluir el tipo de tabla (ver [Tabla R4](#)) y el nombre de la misma como se indica:

Tipo	Nombre de tabla	Código asignando
TE	ciudad	TE_ciudad

Tabla 9. Ejemplo asignación de códigos de tablas en bases de datos.

Atributos: Los atributos deben tener nombres que describan claramente el dato que contiene, las llaves principales deben iniciar por **PK_ nombre de la llave**, las tablas que contengan categorías o tipos, estas deben estar incluidas en los comentarios del campo. Para el caso de las llaves foráneas, estas deben contener en sus nombres **FK_nombre de tabla origen_ nombre tabla destino**

Atributos calculados: El nombre de estos atributos deben iniciar por el prefijo **AC_nombre del atributo**.

NOTA: Es importante resaltar que para cualquiera de la nomenclatura asignada a la **construcción** de bases de datos y sus correspondientes objetos, por ningún motivo se debe utilizar vocales tildadas y la Ñ; el único carácter especial permitido es el guion bajo (_).

PROYECTO:

Identificador: Los proyectos serán identificados por un código que es asignado de forma unívoca por el área de desarrollo, dicho código consta: parte A (código de categoría del proyecto) - parte B (código nombre del proyecto).

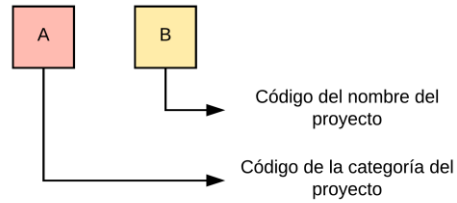


Figura 1. Guía codificación de identificador de proyecto

Nombre del proyecto: Nombre asignado al proyecto por el área de desarrollo que permita contextualizar acerca del mismo; se incluye un código que hará parte del identificador del proyecto.

Nombre	Código
Administrador de propiedad horizontal	APH

Tabla 10. Asignación de código según el nombre del proyecto

Requerimientos: Se deben listar los requerimientos clasificados en dos categorías, funcional y no funcional, y a su vez se le será asignado un código:

Requerimiento	Código asignando
Funcional	F_identificador de proyecto_ #
No Funcional	NF_identificador de proyecto_ #

Tabla 11. Codificación para requerimientos de proyectos.

Versión: Para el control de versiones se hace uso del software GIT bajo la cuenta empresarial de cada uno de los miembros del equipo, el usuario será el código que le fue asignado en el apartado de equipo de trabajo. Así mismo para cada versión se debe tener en cuenta el uso de versiones por número, como se indica:

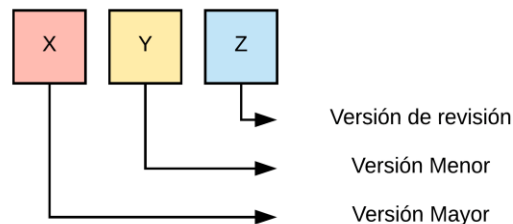


Figura 2. Sistema de versiones para desarrollos de software



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 9

Versión	Ejemplo
Versión mayor X (para cambios significativos)	1.0.0 -> 3.0.0
Versión menor Y (para nuevas funcionalidades)	1.1.0 -> 1.2.0
Versión de revisión Z (indica revisión al desarrollo)	1.1.2 -> 1.2.4

Tabla 12. Ejemplo sistema de versiones para desarrollos de software

ESTRUCTURA DE CARPETAS Y SUBCARPETAS DEL PROYECTO EN DESARROLLO

La estructura de carpetas y subcarpetas deben ser organizadas bajo las siguientes condiciones:

- Se debe respetar la estructura de carpetas fuente y librerías creadas al iniciar un nuevo proyecto por el o los SDK utilizados (Flutter, IONIC y otros), para el caso web los creados por los generadores utilizados (PHP Generator).
- Para el caso del almacenamiento de recursos (imágenes, animaciones, tipos de letras), se deben organizar por categorías (img, tletras) dentro de la carpeta de nombre **assets** en la estructura principal del proyecto.
- Páginas: para el almacenamiento de las diversas páginas que componen el proyecto se debe crear una carpeta con el nombre **pages**. Los archivos almacenados en esta carpeta deben ser nombrados: **nombre del archivo_page**.
- Modelos: para el caso de aplicaciones móviles, es necesario la creación de una carpeta **models** para el almacenamiento de los diversos modelos que se manejen dentro de la aplicación. Los archivos almacenados en esta carpeta deben ser nombrados: **nombre del archivo_model**.
- Proveedores: para el caso de aplicaciones móviles, es necesario la creación de una carpeta **providers** para el almacenamiento de los proveedores encargados de establecer la conexión con los webs services. Los archivos almacenados en esta carpeta deben ser nombrados: **nombre del archivo_provider**.
- Widgets: en el caso del uso del SDK Flutter es necesario la creación de una carpeta llamada **widgets** para el almacenamiento de widgets a los que varias páginas



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 10

acceden. Los archivos almacenados en esta carpeta deben ser nombrados: ***nombre del archivo_widget***.

- Utilidades: se crea una carpeta adicional ***util*** con el fin de almacenar scripts de utilidades en general, necesarias para el proyecto y que son accedidas por varias páginas o proveedores. Los archivos almacenados en esta carpeta deben ser nombrados: ***nombre del archivo_util***.
- Salidas: cada vez que se obtenga el ***release*** o ejecutable de una aplicación se debe ir almacenando en una carpeta diferente a la carpeta del proyecto.

Paquetes y módulos: la organización descrita por módulos o paquetes son parte de las buenas prácticas permitiendo agrupar funcionalidades comunes a varios archivos en uno solo.

Nota Nombres de archivos: los nombres de los archivos deben hacer referencia a la función que desempeña dentro de la aplicación.

Nota Bitácora: Cada vez que se realicen cambios sobre cualquiera de los archivos o carpetas estos deben ser monitoreados por el software GIT y sus comandos básicos, al finalizar el cambio se debe hacer el git commit con el nombre del cambio o versión bajo la cuenta de cada usuario, posteriormente se debe subir al GitHub de la cuenta de la empresa para el proceso de seguimiento y control.

ESTRUCTURA DE CÓDIGO

Archivo fuente: El nombre de la clase principal debe coincidir con el nombre del archivo sin incluir el nombre de la carpeta que lo contiene, adicionalmente debe iniciar por mayúscula. En cada archivo fuente se debe incluir:

- Comentario de inicio que especifica el título de la clase principal, el objetivo de archivo y el copyright (no se especifica fecha de edición, comentarios de cambios, ni nombre de quien lo editó, debido a que esto va por defecto en el seguimiento que se hace desde el commit del Git). Cabe recordar que cada vez que se efectué un cambio significativo se debe realizar el commit del Git.
- Posterior al comentario y respetando la estructura de código, se deben encontrar las importaciones de paquetes con la ruta simplificada, esto considerando que se deba obligatoriamente realizar cambio de nombre de proyecto (***../..../provider/login_provider***).

	Estándar de codificación para desarrollos de software	Código: ECDS-E01
		Versión: 1.0.0
		Fecha: 04-10-2019
		Página: 11

Métodos: Estos se ubican el archivo fuente por orden de acceso desde donde son llamados.

Métodos privados: Es necesario respetar las buenas prácticas definidas para cada lenguaje de programación, pero a modo general los métodos y variables privados serán identificados por un guion bajo al inicio junto con el nombre iniciando con minúscula (`_campoLogin`).

Métodos públicos: Estos se ubican en carpetas específicas para que puedan ser accedidos desde otros archivos (no están ligados a una clase en específica). En general los métodos públicos inician en minúscula asociando el nombre a la funcionalidad que cumple (***alertDialogoCancelar***)

Variables de instancia: Se debe respetar las buenas prácticas de cada lenguaje, se definen al inicio del archivo fuente, si el lenguaje lo requiere se debe especificar el tipo de variable y se debe inicializar la misma a menos de que dicho valor provenga de otro script de código. Los nombres de las variables no deben iniciar por caracteres especiales, deben ser cortos y significativos.

Constantes: Las constantes deben ser declaradas en mayúsculas (si es necesario se debe especificar el tipo de constante) y si es necesario se debe declarar como ***final***.

Sangría: Usualmente cada lenguaje y entorno de desarrollo maneja de forma automática este factor, pero cabe resaltar que se debe respetar la sangría entre líneas de código de diferente nivel (si no se incluyen automáticamente se establecen 2 caracteres como unidad de sangría).

Líneas de código extensas: La longitud de línea no debe superar los 90 caracteres por motivos de visualización, a partir de ellos, si se supera dicha cantidad se debe pasar a la siguiente línea con 4 caracteres como sangría.

Comentarios: Se toman como referencia dos tipos de comentarios, los de implementación y los de documentación.

Comentarios de implementación: estos hacen referencia a la funcionalidad o funcionalidades que ejecutan los métodos, variables entre otras. Sobre cada método o clase externa se debe especificar a modo de título corto la función que cumple. (***/// construye icono logo***), si es necesario describir un proceso de forma más extensa se recurre a los comentarios de bloque (en varias líneas).

Comentarios de documentación: Estos son exclusivos para cuando se realiza el commit en el Git, estos describen cambios en el código, nuevas funcionalidades, versiones, entre otros.



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 12

Declaración por línea: al realizar la declaración de variables, es necesario realizar un comentario de implementación que a diferencia de los métodos no se hace sobre cada método, sino en la misma línea al finalizar la declaración.

Localización: Al declarar variables para el uso de un método en específico, estas deben estar sobre el método al principio de cada bloque posterior al comentario de implementación. No debe haber redundancia en la declaración de variables que ya hayan sido declaradas en un nivel superior.

Declaración de clases: El nombre de la clase debe iniciar por mayúscula, la abertura de la clase “{” debe ir en la misma línea en la que se declaró la clase y el cierre de bloque “}” debe ir en una nueva línea después de finalizado el contenido del bloque.

Sentencias: Las sentencias dentro de las clases o métodos deben respetar la sangría especificada anteriormente.

Espacios en blanco: Se deben dejar espacios en blanco y saltos de línea entre clases, métodos y declaración de variables, esto para mejorar la legibilidad del código en bloques.

Manejo y control de errores: Es estrictamente necesario que para el caso de peticiones POST, GET, o cualquier otro tipo de petición se realice control de errores mediante sentencias como **Throw, Try, Except, Else, Finally** según sea el caso.

Valores de retorno: los valores de retorno deben ser simples y comprensibles en cuestión de nombre, de acuerdo al contexto que manejen.

Tipos de datos que se manejan: Enteros (int), booleanos (bool), cadenas (str), listas (list o array), diccionarios (json -> clave, valor).

Uso de azúcar sintáctico: Hace referencia a el uso de sintaxis diseñada para hacer la construcción de código de programación más comprensible tanto de leer como de expresar, es por ello que se recomienda su uso bajo los términos de cada lenguaje:



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 13

Ejemplo en Dart:

```
builder: (context, constraint) => _widget==null?MainPage():_widget
```

El azúcar sintáctico en este caso se encuentra subrayado con azul, el cual traduce:

```
If (_widget == null) {  
    MainPage();  
} else {  
    _widget  
}
```

Logrando reducir el código a una sola y corta línea.



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 14

TABLAS DE REFERENCIA

Nombre	Código
Gestión, planificación y estrategia	GPE
Desarrollo y nuevas tecnologías	DT
Monitoreo y control de riesgos	MCR
Infraestructura de telecomunicaciones	IT
Atención al cliente	AC

Tabla R 1. Códigos para áreas del centro de desarrollo

Nombre	Código
Levantamiento de requisitos	LR
Gestión de proyecto	GP
Diseño	D
Programación	P
Pruebas y detección de errores	PDE
Cambios	C
Monitoreo	M
Planes de contingencia	PC
Copias de seguridad	CS
Administración de servidores	AS
Administración de redes	AR

Tabla R 2. Códigos para unidades del centro de desarrollo

Nombre	Permisos	Código
Propietario	Configuración, mantenimiento de la base de datos (pueden eliminar bases de datos)	P
Administrador	Creación (crear y alterar), lectura y escritura	A



Estándar de codificación para desarrollos de software

Código: ECDS-E01

Versión: 1.0.0

Fecha: 04-10-2019

Página: 15

Administrador de acceso	Pueden asignar o denegar los permisos a la base de datos.	AA
Visitante	Solo permiso de lectura	V
Copias de seguridad	Permiso para ejecutar las copias de seguridad	CS
Administrador equipo de trabajo	Lectura y escritura	AET

Tabla R 3. Códigos para áreas del centro de desarrollo

Nombre	Código
Tabla Entidad	TE
Tabla de Asociación	TA
Tabla Temporal	TT

Tabla R 4. Códigos para áreas del centro de desarrollo