**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| Programa de formación | Desarrollo multimedia y web |
| --- | --- |

| Competencia | 220501102 - Integrar elementos multimedia de acuerdo con técnicas y herramientas de producción digital. | Resultado de aprendizaje | 220501102-02 - Codificar los elementos del proyecto utilizando herramientas de desarrollo y maquetación según estándares de usabilidad y accesibilidad. |
| --- | --- | --- | --- |

| Número del componente formativo | 011 |
| --- | --- |
| Nombre del componente formativo | Codificación de los elementos del proyecto multimedia |
| Breve descripción | A través del conocimiento de los lenguajes de programación es posible llegar a la construcción de elementos que se harán visibles, por medio de navegadores que permitirán al usuario entender y utilizar las herramientas para solucionar los problemas que se presenten dentro del uso de las interfaces. |
| Palabras clave | Programación, variables, código, entornos de desarrollo, web, SEO. |

| Área ocupacional | 2 - Ocupaciones en ciencias naturales, aplicadas y relacionadas |
| --- | --- |
| Idioma | Español |

1. **TABLA DE CONTENIDO**

**Introducción**

**1. Entornos y herramientas de codificación para el desarrollo de proyectos web**

* 1. Entornos de desarrollo
  2. Entorno de ejecución multiplataforma
  3. Kit de Desarrollo de Software- SDK
  4. Lenguajes de programación
  5. Fundamentos de programación

**2. Herramientas de programación**

**3. Metodologías ágiles de desarrollo**

3.1 Metodologías ágiles de desarrollo y marcos de trabajo para la gestión de proyectos multimedia

3.2 Servicios web

1. **DESARROLLO DE CONTENIDOS**

**Introducción**

Apreciado aprendiz, bienvenido a este componente formativo donde se abordará la codificación de los elementos del proyecto multimedia, por favor ingrese al siguiente recurso para ampliar la información inicial:



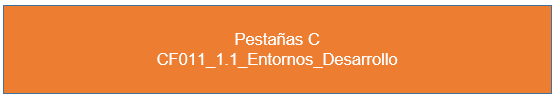
1. **Entornos y herramientas de codificación para el desarrollo de proyectos web**

Entendiendo la codificación y sus capacidades, se puede definir un entorno de desarrollo como el conjunto de elementos, herramientas, leyes y procedimientos para desarrollar un código donde implementar los códigos y caracteres. Para su implementación, es importante conocer desde los entornos de desarrollo hasta los diferentes fundamentos de programación, tal como se observará a continuación.

* 1. **Entornos de desarrollo**

Este término se puede percibir como un Entorno de Desarrollo Integrado (IDE) que es una herramienta de desarrollo, utilizada para escribir, probar y perfeccionar un programa, proporcionando a los desarrolladores interfaces y herramientas comunes al momento de trabajar y desarrollar.

Al momento de elegir un entorno es fundamental conocer el lenguaje con el cual se definirá la programación, tanto en la parte visible o *Frontend*, como en la parte oscura y escondida, conocida como *Backend*. Normalmente el desarrollo de entornos tiene tres niveles, que se clasifican como desarrollo, integración y producción, generalmente conocidos como DSP y que se explican a continuación:



Los anteriores entornos deben cumplir características básicas para que sean atractivas para los usuarios, al igual que facilitar los procesos de programación, permitiendo entornos fáciles para trabajar.

Algunos de los lenguajes de programación más conocidos y usados son:



Una de las características del código JavaScript es que sus etiquetas deben cerrarse:

| <script> </script> |
| --- |

* 1. **Entorno de ejecución multiplataforma**

Es uno de los tantos atributos que tienen y permiten los programas y aplicaciones que hay actualmente en el mercado y el concepto básicamente consiste en implementar y operar en múltiples plataformas informáticas el *software* o aplicaciones que se crean.

Una plataforma es una combinación de *software* y *hardware* que ejecuta una aplicación. Se pueden conocer las plataformas como los sistemas operativos donde van las acciones de los comandos; sin embargo, los dispositivos móviles también pueden ser reconocidos como tal.

Una plataforma de *hardware* podría referirse a la CPU del ordenador y estas pueden ejecutar versiones y sistemas operativos como Windows, Linux, MacOS, etc., cuando se refiere a los teléfonos móviles podemos referirnos a sistemas como Windows, Android e iOS.

Dentro de las plataformas de software más comunes se puede encontrar:

* Android.
* [AmigaOS](https://es.wikipedia.org/wiki/AmigaOS) (m68k), AmigaOS 4 ([PowerPC](https://es.wikipedia.org/wiki/PowerPC)), AROS (x86, PowerPC, m68k), [MorphOS](https://es.wikipedia.org/wiki/MorphOS) (PowerPC).
* [BSD](https://es.wikipedia.org/wiki/BSD), multiplataforma.
* Java.
* Linux.
* iOS.
* Mac OS.
* Microsoft Windows.
* DOS sistemas de tipo en el x86: MS-DOS, IBM PC-DOS, DR-DOS, FreeDOS, etc.

Para que se pueda considerar multiplataforma, el *software* debe ser capaz de funcionar en más de un sistema operativo, el inconveniente es que cada sistema tiene diferentes interfaces y así diferentes lenguajes, lo que hace que sea desgastante. Pensar en un determinado sistema operativo que sea ejecutable en cualquier *hardware* y que funcione correctamente en todas las arquitecturas posibles requiere que el *script* funcione en todos los sistemas operativos, por eso la importancia que el código sea leído por los diversos navegadores y sea funcional en más de una plataforma.

Abordar los problemas que puede llevar la programación multimedia, van desde crear varias versiones para el mismo programa, hasta utilizar aplicaciones que ayudarán a adaptar el código a las plataformas o aplicaciones finales; dependerá mucho de la capacidad de costos, el tiempo para su desarrollo y de los enfoques que se vayan a solucionar al momento de construir código. Cabe resaltar que lo más importante para todas las versiones es reducir a cero los errores, por lo que se debe tener presente la corrección en cada una de las plataformas escogidas.

Depender de algún *software* que permita encontrar esos problemas existentes entre las diferencias de las plataformas o utilizar aplicaciones que permitan construir y ejemplificar la construcción ayudan al momento de visualizar serán herramientas que permitirán ahorrar tiempo al momento de solucionar y reconstruir el código. Aquí se relacionan algunas que pueden ser poderosas herramientas para el desarrollo en multiplataforma.

* **El Cairo:** un *software* libre, que se utiliza para proporcionar un vector basado en gráficos, independiente del API del dispositivo, posee enlaces para muchos lenguajes de programación.
* **Eclipse:** entorno de desarrollo multiplataforma y de código abierto. Funciona en Java con arquitectura que permite configurar diferentes elementos, y soporta herramientas para el desarrollo de *software*. Complementos con Java y C++.
* **FLTK:** otro código abierto multiplataforma. Kit de herramientas, pero más ligero, ya que se limita a la GUI.
* **fpGUI:** un conjunto de herramientas de fuente abierta tipo *widget*, implementado en Object Pascal.
* **GeneXus:** en Windows es una solución rápida de desarrollo de software para la creación de aplicaciones multiplataforma y de despliegue basado en la representación del conocimiento y el apoyo a C#, Cobol, Java incluyendo Android y BlackBerry dispositivos inteligentes, Objective-C para el de Apple dispositivos móviles, RPG, Ruby, Visual Basic y Visual FoxPro.
* **GTK+:** un conjunto de herramientas de código abierto para el *widget* de sistemas tipo Unix con X11 y Microsoft Windows.
* **haXe:** un lenguaje multiplataforma de código abierto.
* **Juce:** un *framework* de aplicaciones escritas en C++, que se utiliza para escribir *software* nativo en numerosos sistemas, sin cambios en el código.
* **Max/MSP:** un lenguaje de programación Visual que encapsula independiente de la plataforma del código con un entorno de ejecución específico de la plataforma en las aplicaciones para Mac OS X y Windows.
* **Mono:** un marco multiplataforma para las aplicaciones y lenguajes de programación.
* **Monocross:** es una fuente abierta del patrón de diseño modelo-vista-controlador en el que se comparten el modelo y el controlador multiplataforma, pero la vista es específica de la plataforma.
* **MoSync:** es un SDK de código abierto para el desarrollo de aplicaciones móviles en la plataforma C++ familia.
* **Framework de la aplicación Mozilla:** una plataforma de código abierto para el desarrollo de Mac OS X, las aplicaciones de Windows y Linux.
* **OpenGL:** una biblioteca multiplataforma de gráficos 3D.
* **Qt (*toolkit*):** una estructura de aplicaciones y Widget Toolkit para sistemas Unix con X11, Microsoft Windows, Mac OS X, disponible tanto en código abierto y como para licencias propietarias.
* **Real Studio:** un IDE RAD desarrollado por Real Software, utiliza un dialecto orientado a objetos del lenguaje de programación BASIC, y produce binarios compilados para MacOS X, Windows y Linux, como también, ser capaz de producir CGI aplicaciones basadas en web.
* **Simple DirectMedia Layer:** una librería de multimedia multiplataforma de código abierto, escrita en C que proporciona acceso de nivel bajo y alto a los gráficos, de entrada, audio, etc.
* **Plataforma Smartface:** una herramienta multiplataforma para Windows que se usa para crear aplicaciones móviles para J2ME, Symbian S60, Blackberry y Android, usando herramientas de arrastrar y soltar y edición de acciones.
* **Ultimate++:** es un marco de desarrollo rápido de aplicaciones multiplataforma de C++ centrado en la productividad de los programadores. Incluye un conjunto de bibliotecas de usuario (GUI, SQL, etc.), Y un entorno de desarrollo integrado. Tiene su propio entorno de desarrollo integrado.
* **wxWidgets:** juego de herramientas *widget* de fuente abierta que es también un marco de aplicación. Se ejecuta en sistemas Unix con X11, Microsoft Windows y Mac OS X. Se permite que las aplicaciones escritas para usarlo para funcionar en todos los sistemas que lo soporta, si la aplicación no utiliza ningún sistema operativo específico de programación, además de a la misma.
* **XPower++:** es un IDE multiplataforma para Windows, Linux, Mac OS X y sistemas operativos móviles.

Dada la complejidad y la cantidad de plataformas se pueden presentar errores los cuales a medida del tiempo se irán depurando, lo importante es construir las características más importantes y relevantes para después transformar los sutiles detalles que ayudarán a que el rendimiento no se vea afectado.

Dentro de las herramientas para el desarrollo en multiplataforma se encuentra Adobe AIR, una de las aplicaciones de la familia Adobe compatible con cualquier sistema operativo y una de las mejores al momento de construir y crear aplicaciones de escritorio a partir del desarrollo de páginas web en HTML, JAVA, AJAX, etc., siendo una de sus características la generación de sus aplicaciones en plataforma multimedia; es decir, lo creado en esta aplicación podrá ser utilizado en cualquier sistema operativo, teniendo en cuenta que:

* Permite al usuario crear su propia aplicación de escritorio de una manera fácil, sencilla y rápida.
* Todos los programas creados tienen opción de ser multiplataforma.
* Se encuentra disponible tanto para los dispositivos móviles inteligente. Sin importar que sistema operativo esté utilizando el mismo.

| AIR es un programa que establece flujos de trabajo compuestos por codificación y compilados en archivos instaladores, su facilidad en la aplicación de lenguajes y formatos permite un flujo más constante y permite detectar errores en los procesos, estos procesos de depurado y solución de errores siguen una serie lineamientos que permite realizar procesos más correctos; es un programa que trabaja de la mano con sus hermanos de la familia Adobe. En general, la creación de las aplicaciones será más eficiente usando las herramientas que proporcionen herramientas a los desarrolladores, para encontrar el sistema y el SDK adecuado, de no ser así la eficiencia de los resultados finales se verán afectados, como trabaja ejecutando la web como una aplicación de escritorio, la posibilidad de trabajar sin internet hace que compañías importantes hayan optado hacer uso de esta posibilidad. | Conoce los primeros pasos con la tecnología de Adobe AIR y una pequeña introducción. |
| --- | --- |

* 1. **Kit de Desarrollo de Software - SDK**

Es una herramienta que permite a los programadores desarrollar programas en lenguajes y programas en concreto para una aplicación determinada. Son elementos que son desarrollados con lenguajes originales para determinados softwares en cuestión, por lo que la mayoría son de uso gratuito.

El desarrollo de un SDK varía dependiendo el fabricante, el lenguaje específico y un sistema operativo dada su finalidad; un SDK viene provisto de muchas herramientas y ventajas dependiendo la necesidad, pero se puede estandarizar que casi todos los SDK deberían tener presentes algunos elementos como los API, una clara documentación de lo que se encontrará en cada SDK: para qué sirve, ventajas y desventajas, etc.; las herramientas de prueba y análisis para la depuración y compilación de sus códigos; editores, bibliotecas y sistemas de tiempos de ejecución, sus entornos de desarrollo, protocolos y los controladores necesarios son los elementos básicos, los cuales son procesos bases para facilitar el trabajo de desarrolladores.

Dado que son herramientas de desarrollo gratuitas es importante que los desarrolladores que las utilicen se comprometan a tener claras unas normas para que sea más sencillo el uso de estas. En el desarrollo de aplicaciones se encontrarán muchas SDK con diferentes licencias de uso. Es importante tener en cuenta esta información al momento de trabajar con SDK y lo siguiente:

* Si un SDK tiene licencia de propiedad, no será apropiado para el desarrollo de *software* de código abierto.
* Si un SDK tiene una licencia tipo GPL, no será apropiado para el desarrollo de aplicaciones de propiedad.
* Si un SDK tiene una licencia tipo GPL, se podría usar partes del código de propiedad, pero el usuario final debe poder modificar la parte del código abierto.

Todas estas aplicaciones de código abierto se están desarrollando continuamente gracias a los kits de desarrollo, los cuales ayudan a programar nuevas funcionalidades para compartirlas a nivel global con otros programadores. Sobre ello, se debe conocer lo siguiente:

| **Video**  **CF011\_1.3\_API\_SDK\_FrameWork\_VideoClase** |
| --- |

Además de los kits de desarrollo de *software* anteriores, se encuentran otros con mayores especificaciones para la programación:



**1.4 Lenguajes de programación**

Los lenguajes están presentes en todas partes y en todas las máquinas que nos acompañan día a día y comprenden lenguajes diseñados para que todas las máquinas y computadoras puedan entender idiomas que puedan comprender entre ellas. Para esto se crea el lenguaje binario del que hemos venido tratando en diferentes temas, dado que estos lenguajes se encargan de convertir funciones e instrucciones en un lenguaje de ceros y unos para que puedan entenderse e interpretarse.

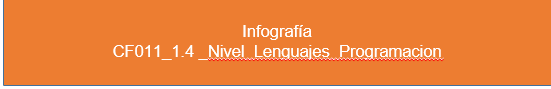
Existen altos y bajos cuando nos referimos a los lenguajes de programación, permitiendo que se logren acciones con pocas palabras al hacer lo mismo que se logra con un lenguaje de bajo nivel.

A continuación, conocerán un poco más de la temática:

| **Video**  **CF011\_1.4A\_LenguajesProgramacion\_VideoClase** |
| --- |

| En los lenguajes de Alto Nivel podemos encontrar a los lenguajes de mayor uso, como los son C++, el Java y el Python, lenguajes que desde la década de 1980 han sido los lenguajes más completos y los más usados por desarrolladores, permitiendo códigos más legibles y más fáciles al momento de su escritura y construcción. | Los lenguajes de programación más utilizados en Big Data - PiperLab |
| --- | --- |

Se pueden agrupar a los lenguajes en diferentes niveles, que son lenguajes orientados a máquinas y la relación que existe entre el *hardware* y el *software*. Tal como se explica a continuación:



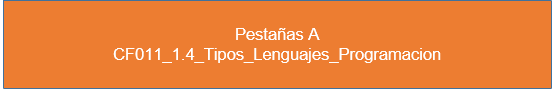
La finalidad y funcionalidad de los lenguajes es el programar; sin embargo, la diferencia es que cada uno tiene un alcance y una forma de comunicación diferentes, algo así como hablar inglés o francés o alemán. Se puede definir que el lenguaje de bajo nivel permite una comunicación con la máquina entendiendo sus códigos de operación y el lenguaje de alto nivel, es introducir datos y códigos en el lenguaje de las máquinas para facilitar instrucciones que se le dan a las máquinas; bajo nivel es un idioma cercado a las máquinas, un alto nivel es un entendimiento e idioma más humano.

Comprendiendo esta diferenciación, se entiende que el conjunto de herramientas permite diferentes acciones como crear, escribir, mantener, depurar, etc.

Algunos *softwares* permiten escribir códigos y crearlos, otros simplemente ingresarlos y ejecutarlos, otros optimizan tiempos y monitorean variables, referencias memorias y ayudan a las correcciones de errores de código. Estos lenguajes permiten la comunicación con el *software* y acá se indica todo lo que se quiere que una página haga, se vea y realice determinadas acciones.

| Todas las acciones que se vean en una página web como los botones, los colores, el diseño, las animaciones, el texto, etc.; es información que se debe recopilar en secciones más pequeñas y traducirlas en instrucciones que la máquina pueda entender. | Las páginas web trabajan en una secuencia de instrucciones traducidas en código, que se trabaja en párrafos o bloques de texto o *script* donde el navegador lee el HTML y otros diferentes lenguajes los interpretan. |
| --- | --- |

HTML y CSS son solo los lenguajes de programación que estructuran las páginas y las informaciones de estilo, dado que HTML era el único que existía en los años 90, pero gracias a otros lenguajes como CSS y JS, estos sitios han venido mejorando. A continuación, se podrán conocer sus características:



Ahora, se relacionan las principales características de los lenguajes:

* **Simplicidad:**el lenguaje debe ofrecer conceptos claros y simples que faciliten su aprendizaje y aplicación, de manera que sea sencillo de comprender y mantener. La simplicidad no significa que se le pueda restar el poder óptimo de funcionamiento.
* **Naturalidad:** se refiere a que su aplicación en el área para la que fue diseñado debe hacerse de forma natural, proporcionando operadores, estructuras y sintaxis para que los operadores trabajen eficientemente.
* **Abstracción:** consiste en la capacidad de definir y utilizar estructuras u operaciones complicadas ignorando algunos detalles. Esto influye en la capacidad de escritura.
* **Eficiencia:** los lenguajes de programación deben traducirse y ejecutarse eficientemente para no ocupar demasiado espacio en la memoria ni gastar mucho tiempo.
* **Estructuración:** permite que los programadores escriban sus códigos de acuerdo con los conceptos de programación estructurada, para evitar la creación de errores.
* **Compacidad:**con esta característica es posible expresar las operaciones con concisión, sin tener que escribir demasiados detalles.
* **Localidad:** se refiere a que los códigos se concentran en la parte del programa con la cual se está trabajando en un momento determinado.

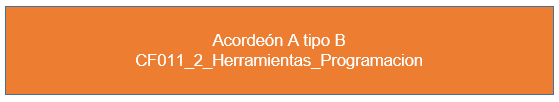
**1.5. Fundamentos de programación**

Son las bases comunes a todos los programas; es decir, es lo primero que debe aprender incluso antes de elegir el programa con el que se quiere programar. A continuación, se relacionan algunos conceptos que posibilitará conocer la intención de las programaciones:



**2. Herramientas de programación**

Permiten realizar aplicativos, programas, rutinas, utilitarios y sistemas para que la parte física del computador u ordenador, funcione y pueda producir resultados. Para comprender mejor el tema se dará un recorrido conceptual por herramientas como Unity y FrameWorks, además de las buenas prácticas de programación:



**3. Metodologías ágiles de desarrollo**

Las metodologías ágiles son un procedimiento opcional en la administración de proyectos que sirven para obtener información de manera más ágil en un proyecto. Estas metodologías analizan el enfoque durante su estructura de creación. En el proceso de implementación de las metodologías ágiles el grupo de trabajo organiza reuniones denominadas *sprint* o iteraciones, estas reuniones ayudan a incorporar la manera de trabajo a la estructura del proyecto, obteniendo una respuesta pronta para organizar el proyecto y a su vez el desarrollo a las condiciones particulares del entorno. Las empresas que implementan estas metodologías pueden administrar sus proyectos de manera adaptable, independiente y eficiente optimizando recursos y aumentando la productividad.

**3.1. Metodologías ágiles de desarrollo y marcos de trabajo para la gestión de proyectos multimedia**

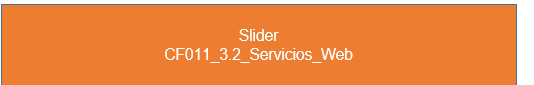
Las metodologías ágiles son adaptables a cualquier proyecto, este procedimiento permite definir y organizar las prioridades en cada etapa de la planificación, a partir de las necesidades y objetivos; y están enfocadas en conseguir resultados desde el comienzo, también ayudan a tener una mayor interacción con el cliente.

Para conocer más sobre el tema, a continuación, se conocerá sobre las metodologías Scrum y Kanban:

****

**3.2. Servicios web**

Es un sistema *software* diseñado para soportar la interacción máquina-a-máquina, a través de una red, de forma interoperable. Para conocer sobre este tema, es importante visualizar el siguiente recurso que explicará el consumo de servicios, XML, peticiones http y *restful*:



1. **MATERIAL COMPLEMENTARIO**

| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| --- | --- | --- | --- |
| 1.4 Lenguajes de Programación | MDN Web Docs. (s.f.). *Referencia de Atributos HTML.* https://developer.mozilla.org/es/docs/Web/HTML/Attributes | Sitio web | https://developer.mozilla.org/es/docs/Web/HTML/Attributes |
| 1.5. Fundamentos de Programación. | Digital Learning. (2019). *Ventanas emergentes y pop up*. [Video]. Vimeo. <https://player.vimeo.com/video/163927259> | Video | <https://player.vimeo.com/video/163927259> |
| 2. Herramientas de Programación | EDTeam. (2018). *FrameWork vs Librería*. [Video]. YouTube. <https://www.youtube.com/watch?v=A-iKX8Shge4> | Video | <https://www.youtube.com/watch?v=A-iKX8Shge4> |
| 3. Metodologías ágiles de desarrollo | BettaTech. (2019). *Scrum y Metodologías Ágiles en ingeniería informática*. [Video]. YouTube. <https://www.youtube.com/watch?v=DT2NEBCAPHw> | Video | <https://www.youtube.com/watch?v=DT2NEBCAPHw> |
| 3.1 Metodologías agiles ágiles de desarrollo | Innovación y estrategia tecnológica en ingeniería UPNA. (2018) *Metodologías ágiles SCRUM y KANBAN.* [Video]. YouTube. <https://www.youtube.com/watch?v=AAZtxEFWqDI> | Video | <https://www.youtube.com/watch?v=AAZtxEFWqDI> |
| 4.1 Base de datos | Edteam. (2020) *¿Cómo diseñar una base de datos? ¿Cuáles son las etapas?* [Video]. YouTube. <https://www.youtube.com/watch?v=WU1tUV_krtA> | Video | <https://www.youtube.com/watch?v=WU1tUV_krtA> |

1. **GLOSARIO:**

| TÉRMINO | SIGNIFICADO |
| --- | --- |
| ASCII | código estándar para el intercambio de información, basado en el alfabeto latino, permite una comunicación entre computadores y humanos. |
| BackEnd | parte interna de la creación de las páginas web, la parte lógica y la capa de código y programación encargada a conectar los servidores y los dispositivos. |
| *Backlog* | lista de trabajo ordenado por prioridades para el equipo de desarrollo que se obtiene de la hoja de ruta y sus requisitos. Los elementos más importantes se muestran al principio del *backlog* del producto para que el equipo sepa qué hay que entregar primero (Atlassian, 2021). |
| C++ | lenguaje de programación enfocado a la manipulación de objetos. |
| *FrameWorks* | o marco de trabajo, es un conjunto de conceptos y prácticas para enfocar una problemática, que sirve como referencia para próximos problemas (Ceolevel, 2019). |
| *FrontEnd* | parte visible de una aplicación o página web; la parte a la que el usuario tiene acceso, desde su diseño hasta su contenido. |
| Kanban | sistema basado en el funcionamiento de los supermercados que reduce la procrastinación en todas las áreas de la empresa (Rock Content, 2019). |
| *Restful* | arquitectura para aplicaciones basadas en redes (como internet), sus siglas significan REpresentational State Transfer y por otro lado RESTful web service o RESTful api, son programas basados en REST. Pero muchas veces se usan como sinónimos (REST y RESTful) (Código naranja, 2021). |
| *Script* | fragmento de código que se encuentra insertado dentro de un documento HTML. |
| Scrum | proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto.  (Proyectos ágiles, 2021). |
| SEO | iniciales en inglés de *Search Engine Optimization* (optimización para motores de búsqueda). Se puede definir como “conjunto de acciones que se realizan para mejorar la visibilidad de un portal de internet en los resultados orgánicos de los distintos buscadores” (Rock Content, 2021). |
| *Sprint* | nombre que va a recibir cada uno de los ciclos o iteraciones que vamos a tener dentro de dentro de un proyecto Scrum (Opern Webinars, 2018). |
| Unicode | set de caracteres universal que define todos los caracteres necesarios para la escritura de la mayoría de los idiomas hablados y que se usan en los computadores (W3C, 2018). |
| XML | siglas de *Extensible Markup Language*, que podemos traducir como Lenguaje de Marcas Extensibles, aunque realmente es un metalenguaje. Normalmente, los archivos XML contienen información de cualquier tipo, ya sea información sensible o información de cualquier ámbito (Alconchel, 2019). |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Atlassian (2021). *El backlog del producto: la lista de tareas pendientes definitiva.* <https://www.atlassian.com/es/agile/scrum/backlogs>

Ceolevel. (2019). *¿Estándares, metodologías o marcos de trabajo? ¿Sabes diferenciarlos?* <https://www.ceolevel.com/estandares-metodologias-o-marcos-de-trabajo-sabes-diferenciarlos>

Código Naranja. (2021). *¿Qué es RESTful Web Service? y ¿qué es REST?* <https://codigonaranja.com/restful-web-service>

Proyectos Ágiles. (2021). *Qué es SCRUM.* <https://proyectosagiles.org/que-es-scrum/>

Opern Webinars. (2019). *¿Qué es XML y para qué se usa?* <https://openwebinars.net/blog/que-es-xml-y-para-que-se-usa/>

Opern Webinars. (2018). *Qué es un sprint de Scrum* <https://openwebinars.net/blog/que-es-un-sprint-scrum/>

Rock Content. (2019). *Conoce el método Kanban para organizar y aumentar la productividad de tu negocio*. <https://rockcontent.com/es/blog/metodo-kanban/>

Rock Content. (2021). *Conoce el método Kanban para organizar y aumentar la productividad de tu negocioSEO: la guía completa para que conquistes la cima de Google en el 2021.*<https://rockcontent.com/es/blog/que-es-seo/>

W3C. (2018). *Codificación de caracteres: conceptos básicos.* <https://www.w3.org/International/articles/definitions-characters/index.es>

1. **CONTROL DEL DOCUMENTO**

|  | Nombre | Cargo | Dependencia | Fecha |
| --- | --- | --- | --- | --- |
| Autor (es) | Eduardo José Velasco Acevedo | Experto Temático | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica | Agosto de 2021 |
| Juan Manuel Reyes Ramírez | Experto Temático | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica | Agosto de 2021 |
| Caterine Bedoya Mejía | Diseñadora Instruccional | Regional Distrito Capital – Centro de Gestión Industrial | Agosto de 2021 |
| Andrés Felipe Velandia Espitia | Revisor Metodológico y Pedagógico | Regional Distrito Capital – Centro de Diseño y Metrología | Agosto de 2021 |
|  | José Gabriel Ortiz Abella | Corrector de estilo | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica. | Octubre del 2021. |

1. **CONTROL DE CAMBIOS**

|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| --- | --- | --- | --- | --- | --- |
| Autor (es) |  |  |  |  |  |