

DE NOVATO A DESARROLLADOR EXITOSO

Los 6 pasos para lograrlo



JULIO LIARTE

DE NOVATO A DESARROLLADOR EXITOSO

Los 6 pasos para lograrlo

JULIO LIARTE

Creado en:



www.escueladeinformaticos.com

SOBRE EL AUTOR



Julio Liarte

- Creador de ESCUELADEINFORMATICOS.com
- Ingeniero en Informática
- Postgrado en Informatización de Empresas
- Postgrado en Negocios Tecnológicos
- Ha trabajado como analista-programador en el sector privado y como técnico de sistemas, jefe de unidad y jefe de servicio en el sector público

Ha acumulado más de 14 años de experiencia desarrollando aplicaciones.

Para más información, visita su blog en
www.escueladeinformaticos.com



ÍNDICE

1. INTRODUCCIÓN
2. ¿QUÉ ES DESARROLLAR APLICACIONES?
3. ¿POR QUÉ DEBO CONVERTIRME EN UN DESARROLLADOR?
4. ¿Y LAS RAZONES PARA NO CONVERTIRSE EN UN DESARROLLADOR?
5. LOS DIFERENTES “MUNDILLOS” DEL DESARROLLO DE APLICACIONES
6. ¿QUÉ LENGUAJE DE PROGRAMACIÓN DEBERÍA APRENDER?
7. ¿POR DÓNDE EMPIEZO? EL MÉTODO DE 6 PASOS
8. ALGUNAS CONSIDERACIONES FINALES

CAPÍTULO UNO

INTRODUCCIÓN

Introducción

Lo primero: ¡Gracias por descargar este libro!

Permíteme que me presente. Me llamo Julio Liarte y, desde hace unos meses, me dedico a ayudar a jóvenes informáticos a que tengan éxito como desarrolladores de aplicaciones (así como a cualquier persona con ganas de aprender), para que puedan ganarse la vida con ello.

En este libro vas a encontrar las respuestas que necesitas para saber cómo puedes convertirte en un desarrollador de aplicaciones exitoso y poder vivir de ello. Te voy a desvelar un Plan para conseguirlo fácilmente en 6 pasos.

Pero vayamos por partes...

Motivación

Una pregunta que te podrías estar haciendo es: "¿Por qué un tipo que desarrolla aplicaciones y se gana bien la vida con ello, escribe un libro gratuito sobre cómo convertirse en un desarrollador de aplicaciones exitoso?".

Como con toda buena pregunta, la respuesta es quizás mucho más larga de lo que probablemente querrías escuchar, pero aquí va:

Porque, sinceramente, estoy completamente convencido de que se trata de la única rama de esta profesión (la de informático) con el potencial suficiente para LIBERAR a las personas, brindándoles la oportunidad de cambiar sus vidas (si así lo desean), y llevándoles tan lejos como ellos quieran. Me explico:

Últimamente, cuando observo el entorno de la gente joven que se dedica a la Informática (o va camino de hacerlo), únicamente veo tres situaciones:

- La del **inocente**, que un día terminó sus estudios (complicados, como lo son en esta profesión), consiguió un trabajo y creyó que "de ahí, al cielo...", pero un día le pusieron de patitas en la calle porque su empresa le dijo que tenía que recortar gastos y, claro, él era de los últimos en llegar... Así que ahora está en paro y sin demasiadas opciones.
- La del **afortunado**, que también terminó un día sus estudios y consiguió un trabajo... y ahora tiene que "pagar por ello", aguantando situaciones de estrés (MUCHO estrés), jornadas interminables en las que además tiene que llevarse trabajo para casa (iy nada de cobrar horas extra!), aguantar reproches, impertinencias, cambios de última hora... y todo por un mísero sueldo (que ni siquiera le da para vivir) y una constante amenaza de convertirse en el próximo inocente.
- La del **iluso**, el estudiante que está formándose y realizando sus estudios con ilusión, con la esperanza de poder algún día acceder al mercado laboral... y sin embargo, lo que esta sociedad le tiene preparado es un índice de paro juvenil brutal, una etiqueta de "generación perdida", una fuga de talento a otros países, etc., etc.

Y todo esto está pasando en plena ***Era de la Información***, en un momento en el que Internet nos está brindando cada día nuevas oportunidades, en el que cada vez son necesarios menos recursos para hacer grandes cosas, en el que la globalización y deslocalización están cambiando las formas tradicionales de hacer negocios (...).

Realmente me entristece mucho cuando conozco casos de gente que está verdaderamente desesperada porque llevan mucho tiempo en paro y no consiguen encontrar un trabajo; o aquellos que sí lo tienen, aguantan situaciones insoportables con salarios precarios, jornadas interminables... y todo esto para satisfacer ¡los intereses de otros!!

¿Quieres saber una cosa?: Afortunadamente, ¡el mundo ha cambiado! Ahora tienes todos los medios a tu alcance (algo impensable en el siglo pasado) para poder ganarte muy bien la vida sin tener que soportar todo esto. Si te dedicas a la informática (o tienes pensado hacerlo) te adelanto que la manera de lograrlo pasa por aprender a desarrollar aplicaciones.

Si nunca lo has hecho o no tienes ni idea de en qué consiste ni por dónde empezar, no te preocupes. En este libro voy a compartir contigo un método sencillo, basado en 6 pasos, para que consigas convertirte en un desarrollador de aplicaciones exitoso y puedas tomar las riendas de tu vida.

CAPÍTULO DOS

¿QUÉ ES
DESARROLLAR
APLICACIONES?

¿Qué es Desarrollar Aplicaciones?

Si nunca has desarrollado aplicaciones antes, este tema probablemente te suene a chino y te parezca algo muy complicado. Y sinceramente, no me extraña. Cuando le digo a la gente que me dedico a desarrollar aplicaciones suelo comprobar cómo se les ponen los ojos en blanco y se produce un breve silencio rotundo (de esos con sonido de grillos de fondo), seguido de un rápido cambio de tema.

Pero, en realidad, no es tan complicado como parece. El **desarrollo de aplicaciones** (también conocido como: programación, codificación, desarrollo de software... –aunque realmente hay diferencias sutiles entre cada término–) , es el proceso de escribir instrucciones (órdenes) que un ordenador pueda ejecutar de forma automática. Por ejemplo:

- Si alguna vez has definido una macro en *Excel*, se podría decir que ya has programado.
- Si alguna vez has diseñado una página *HTML*, se podría decir que ya has escrito código.
- Y si alguna vez has escrito código en un lenguaje de programación convencional como *BASIC*, *Pascal*, *C*, *Java*, *JavaScript* o *PHP*, entonces sin duda se podría decir que ya has hecho tus pinitos en el mundo de la programación.

¿Por qué es necesaria la Programación?

Podrías estar preguntándote: "¿pero para qué el mundo necesita todo esto de la programación?" Y la respuesta es bien sencilla: Se necesita la programación porque los ordenadores no son muy inteligentes (sí, sí, como lo oyes...). Nadie pone en duda las enormes capacidades de cálculo, ejecución, etc. de los ordenadores de hoy en día; pero sin alguien que les diga en algún momento lo que tienen que hacer, no son más que un pedazo de chatarra con algunas luces intermitentes. A toda esta parte "tangible" del ordenador (placas electrónicas, componentes electrónicos, cables...) se le conoce como **hardware**). El **software** es lo que se ejecuta sobre el hardware, la parte "intangible", abstracta... pero sin duda, la parte más interesante de la Informática (y donde hay mayores oportunidades para todos).

Un programa de ordenador, también conocido como "aplicación", o "software", es un conjunto de instrucciones (o "código"), escrito por un programador (o por varios), y que después es ejecutado en un ordenador.

Para no tener que utilizar el conjunto de instrucciones que nos proporciona el fabricante de cada modelo de ordenador concreto (también conocido como "lenguaje ensamblador", o "código máquina), algo que sería una auténtica locura que rozaría casi la paranoia, dada la amplia gama de modelos de ordenador que existen hoy en día y la velocidad con la que cambia la tecnología (por no hablar de las consecuencias psicológicas que podría dejarnos...), se inventaron los **lenguajes de programación**, que se encargan de traducir un conjunto de instrucciones, escritas en un lenguaje "más similar" y próximo a la manera de expresarnos que tenemos los humanos, en las instrucciones que realmente entiende el ordenador (es decir, transforman instrucciones escritas en un lenguaje "entendible" por los humanos a un lenguaje más próximo a la máquina).

Existe una gran variedad de lenguajes de programación hoy en día, tales como: *C*, *C++*, *Java*, *Visual Basic*, *C#*, *Perl*, *PHP*... y un largo etcétera. Hay literalmente cientos (quizás miles) de lenguajes de programación; cada uno con un propósito concreto y destinados a un público determinado. Pero tan sólo unos pocos son ampliamente utilizados en las distintas épocas. Algunos lenguajes de programación se ponen de moda y son utilizados durante algunos años, pero luego son reemplazados por los nuevos lenguajes que van apareciendo. Sin embargo, hay otros lenguajes de programación que perduran durante décadas (*COBOL*, *C* y *C++* son algunos ejemplos).

Los ordenadores sobre los que se ejecuta este software pueden ser de muy distinto tipo. Podrían ser ordenadores personales de escritorio, servidores web disponibles en Internet, u otros ordenadores especializados y menos conocidos que existen en coches, ascensores o en tu horno microondas.

Microsoft Word y todo el paquete ofimático de *Microsoft*, sería un ejemplo de un programa que se ejecuta en muchos de nuestros ordenadores de escritorio (y portátiles). El sitio web de *Amazon.com* es un ejemplo de un programa (un software llamado *Obidos*) que se ejecuta en miles de servidores web de *Amazon* en todo el mundo.

Pero es mucho más que sólo programar

Mientras que la programación podría ser definida como el acto de "codificar" o, lo que sería lo mismo, escribir instrucciones de ordenador en un editor utilizando un lenguaje de programación, ser desarrollador de aplicaciones, en cambio, no va solo de escribir código.

El **desarrollo de software** (un término del que probablemente hayas oído hablar y que utilizan indistintamente para referirse a la **Programación**), es un término más general y que abarca el proceso de pasar del concepto (algo abstracto) al producto terminado (algo concreto). Mientras que la programación es el acto de "mecanografiar instrucciones", el desarrollo de software incluye:

- Hablar con los usuarios (o usuarios potenciales) de la aplicación para entender sus requisitos y generar nuevas ideas de funcionamiento.
- Elaborar documentos y diagramas (serían como los planos para un arquitecto) con las especificaciones que describen la funcionalidad de la aplicación.
- Decidir las características y las cuestiones de diseño (aspectos más técnicos) con el resto del equipo de desarrolladores.
- Escribir el código (entiéndase "*programación*").
- Probar el código.
- Corregir los errores.
- Preparar el software para su distribución.
- Instalar el software en el entorno real en el que será utilizado por los usuarios.
- Etc.

Mientras que la programación es la parte más divertida del trabajo para muchos programadores, se requieren varios pasos antes y después de que uno se siente en su mesa y se disponga a escribir miles de líneas de código.

Me parece interesante la analogía con el mundo de la construcción inmobiliaria. ¿Verdad que sería absurdo pensar que levantar un edificio se resume en poner los ladrillos? Hay todo un trabajo previo de diseño y arquitectura, de estudios de materiales y cargas, de planificación de recursos y tiempo, distintos trabajos de construcción (electricidad, fontanería, albañilería, pintura...), solicitud de licencias, etc., etc.

CAPÍTULO TRES

¿POR QUÉ DEBO
CONVERTIRME EN UN
DESARROLLADOR?

¿Por qué debo convertirme en un desarrollador?

Muy sencillo: porque hay trabajo para todos, porque vas a adquirir muchas habilidades técnicas interesantes, porque vas a poder optar a un sueldo digno, porque existen opciones de flexibilidad en el trabajo, porque supone un desafío, porque sentirás una sensación muy gratificante al crear algo tú mismo... y porque mola mucho!

Trabajo para todos

Según el Departamento de Trabajo de Estados Unidos, 8 de las 10 ocupaciones de más rápido crecimiento entre 2000 y 2010 estuvieron relacionadas con la Informática. Si echamos un vistazo a los portales de empleo, podremos comprobar que siempre aparecen varios puestos de trabajo relacionados con la Informática entre los 10 primeros. Pero, ¿cómo es posible que haya tantas ofertas de empleo relacionadas con el desarrollo de aplicaciones informáticas (incluso en años de crisis económica)?, ¿será algún tipo de moda pasajera?

La respuesta es "no", no es ninguna moda pasajera y esto es sólo el comienzo. En plena *Era de la Información* y en una sociedad cada vez más dependiente de la tecnología y que se apoya en ésta para cualquier proceso cotidiano (lo vemos a diario en las gestiones con el banco, en los trámites con las Administraciones Públicas, en los envíos que hacemos por correo, en las compras en cualquier comercio, en las reservas de vuelos y hoteles, etc.), va a ser cada vez más difícil (sino imposible) prescindir de las personas que tienen este conocimiento.

Te lo voy a demostrar. Vayamos a la web de [Dice.com](#) (portal americano de búsqueda de empleo relacionado con la informática y la tecnología), y hagamos una búsqueda con el término "Developer" (desarrollador, en inglés). En el momento de escribir estas líneas, el buscador arrojó un resultado con más de 28.500 ofertas de empleo.

Un claro ejemplo es un amigo mío que es desarrollador de aplicaciones .NET y trabaja actualmente en Los Ángeles. Activó su perfil en [Dice.com](#) hace unos meses y recibió 40 llamadas telefónicas en tres días.

Sé lo que estarás pensando: "ya, ya... pero eso es en Estados Unidos... seguro que en otros países no es igual". Muy bien, hagamos la prueba. Probemos con España (uno de los países más golpeado por la crisis en los últimos años). Vayamos a la web [Infojobs.net](#) y hagamos la misma búsqueda con el término "Programador". En el momento de escribir estas líneas, el buscador arrojó un resultado con más de 2.500 ofertas de empleo. No está nada mal para tratarse de España (país con un 25% de paro) en un día cualquiera.

Créeme, si te dedicas a esto nunca te va a faltar trabajo. En toda mi carrera laboral como desarrollador, nunca me ha faltado. Y esta es sólo una opción: la de trabajar como empleado asalariado en alguna empresa u organización; pero en esta profesión existen otras muchas posibilidades: trabajar por cuenta propia como *freelance* ofreciendo tus servicios a particulares u empresas o, incluso, atreverse con algún tipo de proyecto de emprendimiento en Internet, o desarrollar alguna App y cobrar royalties por la propiedad intelectual... El abanico es muy amplio, por lo que las posibilidades se multiplican en este sector.

Es así de sencillo: si se diese la remota y más que lejana posibilidad de que no encontraras trabajo... ¡te lo creas tú mismo!

Habilidades técnicas

Cuando alguien en otras áreas (ventas, relaciones públicas, administración...) busca un trabajo, a menudo le lleva meses encontrar un puesto. Cuando un desarrollador con conocimientos actualizados busca trabajo, con frecuencia tiene una entrevista a los pocos días. Esto es en parte debido a la abundancia de ofertas de trabajo de desarrollador (como ya hemos comentado), pero también se debe al hecho de que constituye una "habilidad difícil", compleja o dura ("*hard skills*", que dirían en inglés).

Estas habilidades difíciles son las habilidades como la programación, el diseño, la ingeniería... actividades que constituyen el núcleo de cualquier negocio. Las habilidades "fáciles" ("*soft skills*") permiten realizar actividades como ventas, administración, relaciones públicas... Ambas son valiosas y necesarias pero las habilidades "*hard*" son más sencillas de plasmar en un currículum vitae (de probar y cuantificar), por lo que el proceso de búsqueda de empleo es a menudo mucho más rápido para los desarrolladores que para un comercial.

Te garantizo que si envías tu currículum a cualquiera de los principales portales de empleo, con alguna habilidad del tipo: "*NET developer*", "*Senior Java Developer*", o "*Senior PHP*" en el título, seguramente recibirás tu primera llamadas en unos pocos días.

Sueldo digno

En mi opinión, nunca se debe tomar un trabajo simplemente por el sueldo. Pero si te gusta el desarrollo de aplicaciones (o piensas que puedes disfrutar con ello), y alguien te va a pagar más que otro por hacerlo... quizás no sea tan mala idea prestarle algo de atención.

Es cierto que el sueldo no es igual en todos los países. Suele ser mayor en los países en los que las empresas producen algo y comercializan sus propios productos. En países como España, donde la mayor parte de las empresas vinculadas a este sector se dedica únicamente a ofrecer sus servicios (servicios que, a su vez, subcontrata a otra empresa...), la situación es bien diferente. Pero, como te he explicado antes, si no te gusta trabajar como empleado en estas empresas, tienes muchas otras opciones si te dedicas a esto.

En cualquier caso, merece la pena que nos tomemos un minuto y vayamos a la web [Salary.com](#) (web con datos e información sobre salarios en Estados Unidos), e introduzcamos el término "*Programmer I*" y una ciudad en el asistente (no se trata de una nomenclatura muy común, ya que utilizan "*Programmer I*" para referirse a los programadores con 0-3 años de experiencia, "*Programmer II*"... así hasta el "*Programmer V*", para referirse a desarrolladores con mayor experiencia y rango).

En mi caso, realicé una búsqueda en Boston, en la que los programadores con este nivel aparecen listados con sueldos entre 46.000\$ y 75.000\$, con la mayoría en torno a los 60.000\$. No está nada mal para 0-3 años de experiencia, ¿eh?

No es nada raro encontrar sueldos de 125.000\$-150.000\$ como desarrollador *senior* en una ciudad mayor (por supuesto, esto requeriría 5-7 años de experiencia y conocimiento de nivel *Senior* de una tecnología específica).

Flexibilidad

Cada vez es más frecuente encontrar casos de gente que trabaja desde su casa. A menudo se ofrecen puestos de trabajo como desarrollador de aplicaciones que ofrecen flexibilidad horaria y descentralización (posibilidad de trabajar desde tu casa, desde una cafetería o desde otra ciudad).

A un desarrollador motivado le resultará mucho más sencillo avanzar más en los proyectos desde su casa que desde la oficina, debido a la falta de interrupciones (aunque requerirá de una cierta disciplina de trabajo para evitar las distracciones). El progreso también es bastante fácil de medir por parte de los directivos, por lo que éstos no suelen poner impedimentos para que los desarrolladores trabajen desde casa, por lo menos una parte de su jornada semanal (aunque todo esto dependerá por supuesto de cada país y de su cultura del trabajo y del esfuerzo).

Hoy en día, las limitaciones geográficas no existen. Todo lo que necesita un desarrollador para hacer su trabajo es un ordenador portátil y una conexión a Internet. De este modo, podrías escribir miles de líneas de código desde cafeterías, desde las montañas e incluso desde la playa.

Supone un desafío

Si bien las razones que doy más arriba pudieran resultar atractivas para alguien que lo vea desde fuera, una vez te conviertes en un desarrollador valoras más otros aspectos quizás más relacionados con el trabajo bien hecho y la superación.

Como en todas las profesiones, hay momentos buenos y momentos malos. Habrá veces en las que te encuentres con algún problema o error que se te resista y te haga pasar quizás 10 horas perseverando para dar con la solución... en esos momentos tiene verdaderamente que gustarte esta profesión y disfrutar de tu trabajo para no volverte loco (y no importa tanto la cantidad de dinero que ganes).

Si conoces a algún desarrollador de aplicaciones, pregúntale si disfruta de su trabajo. Probablemente te contestarán algo parecido a "es un desafío y estoy constantemente aprendiendo cosas nuevas".

El reto consiste en solucionar problemas. Algunos problemas serán tan simples como "¿cómo podemos ofrecer a este cliente un nuevo botón en esta página para que puedan imprimir desde aquí?". Otros serán tan complejos como "tenemos todo un proyecto que nos llevó 3 años de programación en *Visual Basic* y queremos migrarlo a *Java*. ¿Qué hacemos?".

Lo que sí te puedo asegurar es que al final, cada persona que conozco que se dedica al desarrollo de aplicaciones (y que disfruta de su trabajo), ama la solución de problemas.

El simple hecho de crear

Voy a ponerme un poco filosófico (te pido disculpas). Existe un gran debate sobre si el desarrollo de aplicaciones es un arte o una ciencia. Personalmente, creo que tiene un poco de ambos. Aunque esto no es para mí lo importante. Lo verdaderamente importante es el torrente de endorfinas que se siente cuando después de programar un código, lo ejecutas y lo pruebas por primera vez. La primera vez que una aplicación web que has programado tira de registros de una base de datos y los muestra en pantalla, perfectamente tabulados... produce una estupenda sensación de logro.

Suena estúpido (me doy cuenta de ello), pero el hecho de haber creado algo "tangible" (por ejemplo, un sitio web) es quizás la mayor razón por la que no he dejado de desarrollar aplicaciones durante los últimos 10 años. Crear algo de la nada, simplemente con tus propios recursos y conocimiento, es una gran sensación. Es como cuando éramos niños y nos gustaba juntar piececitas para construir un enorme castillo que, a su vez, utilizaríamos para... Con esto pasa lo mismo. La sensación de construir una aplicación que resuelva un problema concreto es muy gratificante.

Es a veces... ¡guay!

Vale, vale, la programación no se considera generalmente "guay". Muchas veces quizás te encuentres desarrollando una aplicación de facturación de uso interno que sólo unas pocas personas van a utilizar. Aunque no es tan aburrido como suena, reconozco que no es *Disneyland* tampoco.

Pero también hay casos de gente que desarrolla el sitio web de un conocido festival de cine, o que trabaja para *Google* o *Facebook* en alguna nueva característica, o que desarrolla un sitio web para una empresa que vende millones de dólares al año a través de su web, o el tipo que desarrolla una *app* con 16 años y la vende a algún gigante tecnológico por varios cientos de miles de dólares... En esos momentos sí es divertido ... y sí, tal vez incluso podría considerarse que es bastante guay el ser desarrollador de aplicaciones.

CAPÍTULO CUATRO

¿Y LAS RAZONES
PARA NO
CONVERTIRSE EN UN
DESARROLLADOR?

Si no te gusta la Programación

Si eres un buen desarrollador (o te quieres dedicar a esto), seguramente habrá momentos en que vayas a estar rodeado de código 40 horas a la semana (y probablemente, pensando en él también los fines de semana). Si no te gusta programar, sinceramente este trabajo no es para ti.

Tienes que actualizar constantemente tus habilidades

La mayoría de los lenguajes de programación y tecnologías cambian cada 2-3 años (esto no para inunca!). Si no tienes una sed de aprendizaje permanente o si te cuesta adaptarte a los cambios del entorno, el desarrollo de aplicaciones no es para ti.

Los buenos desarrolladores son aprendices de por vida y siguen leyendo libros y formándose constantemente.

La gestión en las empresas

Es difícil encontrar buenos gerentes en cualquier área de una empresa, pero parece que la situación es especialmente mala cuando nos referimos al área de *Tecnologías de la Información (IT)*.

En muchos casos, los gestores de *IT* fueron un día "tecnólogos", por lo que están más orientados hacia la tecnología que a dirigir y gestionar equipos de personas. En ninguna parte he visto equipos peor gestionados que en esta área. En otros casos, los gestores no tienen ni idea de tecnología (y créeme que no sabría decirte qué situación es peor), por lo que se cometen muchos errores de estimación y planificación en los proyectos que después se traducen en enormes cargas de trabajo, estrés y frustración.

Así que si esperas trabajar como empleado en una empresa desarrollando aplicaciones, lamento comunicarte (y ojalá me equivoque) que es muy probable que des con un pésimo gestor de proyecto (o incluso de departamento). La buena noticia es que, como ya te comenté antes, si no te gusta lo que hay siempre puedes montártelo por tu cuenta.

Las horas extraordinarias

Esto varía de una situación a otra y de una semana a otra pero, dado que los proyectos suelen tener unos plazos y unas fechas límite determinadas, en algún que otro momento tendrás que trabajar horas extra.

Las horas extra en esta profesión casi nunca se pagan. Si trabajas en una empresa, rara vez te pagarán las que tengas que dedicar en la oficina (y ni mucho menos las que suponen el trabajo que te lleves a casa). Si trabajas por libre como *freelance* o emprendedor, está claro que no (aunque quizás en estos casos lo lleves de otra manera).

Mi recomendación es que si se trata de algo puntual, lo aceptes como parte de esta profesión (gajes del oficio). Pero si la cosa se convierte en algo continuo y frecuente, mejor cambia de trabajo.

CAPÍTULO CINCO

LOS DIFERENTES “MUNDILLOS” DEL DESARROLLO DE APLICACIONES

Los diferentes “mundillos” del desarrollo de aplicaciones

No todos los desarrolladores de aplicaciones realizan el mismo tipo de trabajo. Podríamos distinguir los siguientes seis "mundillos" del desarrollo de software:

Productos

Esto incluye trabajar para una gran compañía como *Microsoft*, desarrollando sus aplicaciones *Word* o *Excel*; para una empresa como *Google*, desarrollando *Google Maps* o *Gmail*; o para una empresa más pequeña como *Salesforce.com* desarrollando su aplicación web.

El desarrollo de productos es extremadamente complejo, a menudo con plazos ajustados y una gran cantidad de horas extras antes de un lanzamiento. Me atrevería a decir que los desarrolladores con más talento se mueven en este mundillo, ya que les ofrece los mayores retos y les permite construir un software de mayor complejidad técnica y calidad, trabajar con los mejores profesionales y, como decía Steve Jobs, desarrollar productos que cambien el mundo.

Desarrollo Corporativo

Normalmente, esto se parece bastante a trabajar para un banco, compañía de seguros u otra gran corporación, desarrollando aplicaciones para su departamento de contabilidad, call-center, departamento de envíos, etc.

En este escenario tendrás que trabajar con tecnologías empresariales como .NET o Java, y construir aplicaciones web, aplicaciones de escritorio (normalmente aplicaciones .NET Windows), o aplicaciones móviles.

Yo he sido un desarrollador corporativo durante varios años de mi carrera. Aquí es donde se encuentran la mayoría de las ofertas de empleo como desarrollador.

Software “embebido”

Esta categoría incluye el software que se ejecuta en nuestros coches, en los controladores de los ascensores y en los dispositivos GPS que utilizamos, por poner algunos ejemplos.

Es un mundo que está a medio camino entre el hardware y el software. En este tipo de software sólo se tiene una oportunidad para hacer las cosas bien (no hay re-ediciones de versiones que corrijan errores), por lo que los ciclos de lanzamiento y las pruebas son más largas.

Escribir software embebido es bastante más complejo que desarrollar cualquier otro tipo de aplicaciones. Los sueldos seguramente sean superiores, pero las oportunidades de trabajo son mucho más limitadas que con el desarrollo corporativo.

Desarrollo de videojuegos

Si te gustan los videojuegos, el desarrollo de uno de ellos es una maravilla. Aunque supone también una gran cantidad de trabajo.

Los relatos de primera mano que he escuchado de desarrolladores de videojuegos indican que les apasiona el trabajo, pero odian las largas semanas (60-80 horas). El sueldo también tiende a ser más bajo que el desarrollo corporativo, lo cual tiene mucho sentido: si eres un desarrollador corporativo y estás desarrollando una aplicación de facturación, y además te apasionan los videojuegos, ¿no aceptarías una reducción salarial con tal de trabajar en uno de ellos?

El desarrollo de videojuegos requiere de una mente matemática, y es la única área del desarrollo donde dicen que las matemáticas a nivel universitario son prácticamente una necesidad.

Consultoría

También podrías trabajar para una gran empresa de consultoría como *EDS*, *BearingPoint*, o *Accenture...*, o para una pequeña empresa de consultoría (cosa que hice durante un año). La consultoría puede ofrecer bastante diversión y se tiende a trabajar con las tecnologías de vanguardia y las nuevas técnicas de programación. Pero trabajar para las grandes empresas de consultoría requiere una gran cantidad de viajes y las más pequeñas, probablemente, no van a tener trabajo a tiempo completo para ti.

El sueldo tiende a ser comparable o mayor al del desarrollo corporativo. Aunque existen algunas empresas de consultoría que son una auténtica aberración para esta profesión y que con sus modelos de negocio, basados en la subcontratación y la rotación de personal, están jugando con las ilusiones y expectativas de miles de jóvenes que un día cayeron atrapados en sus redes y que ahora se encuentran soportando situaciones esperpénticas a cambio de unas condiciones nefastas.

Personalmente, el mundo de la consultoría me parece un área que puede resultar interesante (y quizás la elección natural) para transitar de ser un desarrollador corporativo a montar tu propio negocio. Pero mi recomendación es que te informes muy bien de la empresa en la que te vas a meter (ya que como suele decirse: no todo el monte es orégano).

Freelance

Una vez se tenga experiencia y contactos, se podría plantear el hacer proyectos por tu cuenta, y transitar a un desarrollador *freelance*. El trabajo está bien pagado (el sueldo no será un problema), pero la parte complicada será conseguir mantener un flujo constante de trabajo (una cosa es que cuando te salga un trabajo éste esté bien pagado, y otra muy diferente es que te salgan esos trabajos con la frecuencia suficiente).

Como todo, tiene cosas buenas y cosas malas. Por un lado, tienes la oportunidad de trabajar para ti mismo y de tener toda la libertad que supone ser tu propio jefe. Pero, por otro lado, tienes que hacer frente a tu propio seguro de salud, no se te pagarán los días de vacaciones, tienes que manejar tu propio marketing, ventas, facturación, ahorrar para la jubilación...

Pero, sin duda, se trata de una meta muy atractiva y alcanzable para muchos desarrolladores con mente emprendedora (entre los que me incluyo yo también).

CAPÍTULO SEIS

¿QUÉ LENGUAJE DE
PROGRAMACIÓN
DEBERÍA APRENDER?

¿Qué lenguaje de programación debería aprender?

Esta es una cuestión un tanto subjetiva. Sería algo así como preguntar "¿Debo aprender español, francés o alemán si quiero ser intérprete? Es difícil responder sin saber algo más sobre qué trayectoria te gustaría seguir una vez te conviertas en un desarrollador.

Debido a que es muy probable que estés leyendo este ebook sin apenas conocimientos sobre lenguajes de programación, no quiero arrojarte un listado de 25 posibles lenguajes para aprender.

Existen grupos de lenguajes basados en los distintos "mundillos" del desarrollo de software que ya hemos comentado más arriba. En general (y estoy haciendo generalizaciones y seleccionando los lenguajes de programación más populares, simplemente para reducir la confusión del lector):

- ❑ Los productos de escritorio son típicamente programados en *C++, C#, VB.NET, o Java*.
- ❑ Las aplicaciones web son típicamente programadas en *ASP.NET, PHP, Ruby on Rails, o Java*.
- ❑ El desarrollo corporativo típicamente utiliza cualquiera de los lenguajes de escritorio o web (o ambos).
- ❑ El software "embebido" típicamente utiliza una versión especializada de *C* o *C++*.
- ❑ El desarrollo de videojuegos típicamente utiliza *C++* (y más recientemente, *C#*).
- ❑ La consultoría y el trabajo *freelance* tiende a ser realizado en los lenguajes de escritorio y web.

Puesto que existe también una gran cantidad de lenguajes entre las fronteras de estos mundillos, voy a suponer que una de tus mayores prioridades es conseguir ganarte la vida como desarrollador de aplicaciones y que deseas invertir tu tiempo en aprender un lenguaje de programación, con una cierta garantía de que seguirá utilizándose en los próximos años. Siguiendo esta lógica, y la opinión de muchas personas en la industria del software (entre las que me incluyo), puedo comentarte que el desarrollo de aplicaciones de escritorio está muriendo. Todavía no está muerto del todo (aún quedan ciertos nichos en los que se demandan este tipo de desarrollos) pero, en general, los lenguajes de programación web son la tendencia presente y futura, así como lo es también el desarrollo de aplicaciones móviles (*apps*).

Con esto en mente, aquí te dejo la lista de lenguajes de programación que recomendaría:

ASP.NET

ASP.NET es el primer lenguaje de programación que te recomendaría (junto con **C#**, para ser más específicos). En realidad, **ASP.NET** es lenguaje de programación destinado al desarrollo de aplicaciones web y que se apoya en otro lenguaje de propósito general como es **C#**.

ASP.NET se ejecuta en un entorno *Windows* y las aplicaciones web que desarrollemos tendrán que ser desplegadas en un servidor web llamado **Internet Information Server (IIS)** (incluido en cualquier sistema operativo *Windows*).

Se utiliza un entorno de desarrollo llamado **Visual Studio** para programar, y típicamente también un sistema de bases de datos como **SQL Server** (existen versiones "Express" de ambas herramientas, que están disponibles de manera gratuita).

Sin duda, esta es la ruta que recomendaría a cualquiera que quisiera iniciarse en el desarrollo de aplicaciones, principalmente porque es muy sencillo empezar a desarrollar si se tiene un equipo con *Windows* (casi todo el mundo lo tiene), y porque te permitirá trabajar para las grandes empresas, para las pequeñas empresas, para las empresas de productos web o para ti mismo (como *freelance* o emprendiendo tu propio proyecto). Además, hay muchas ofertas de empleo que requieren de estos conocimientos y se trata de una de las plataformas de desarrollo más populares del mundo.

En resumen, con **ASP.NET** podrás desarrollar aplicaciones web que serán ejecutadas sobre una máquina con *Windows* (normalmente, un servidor).

PHP

PHP es un lenguaje de "scripting" de código abierto. ¿Y esto qué significa? Pues que fue uno de los primeros lenguajes de programación para el desarrollo de aplicaciones web que se podían incorporar directamente dentro del documento *HTML* en lugar de llamar a un archivo externo que procesara las instrucciones. El código *PHP* es interpretado por un servidor web, que tiene un módulo de procesado de *PHP*, y que genera la página Web resultante.

Es ideal para el desarrollo de aplicaciones web utilizando herramientas de software libre. PHP se ejecuta en un servidor web **Apache** (disponible para cualquier sistema operativo) y, normalmente, utiliza un sistema de bases de datos **MySQL** (ambas plataformas son de código abierto).

Se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hoy en día. Por ello ha atraído el interés de múltiples sitios con gran demanda de tráfico como *Facebook*, para optar por *PHP* como tecnología de servidor.

Hay muchas ofertas de empleo que requieren conocimientos de *PHP*, aunque no suelen estar tan bien remuneradas como las de *.NET*. Como dato adicional, que sepas que las empresas son generalmente más propensas a utilizar *.NET* o *Java*, mientras que las *startups* lo son más a utilizar *PHP* o *Ruby on Rails*.

En resumen, con *PHP* podrás desarrollar aplicaciones web que podrán ser ejecutadas sobre cualquier máquina (con cualquier sistema operativo), y utilizando herramientas de software libre.

Ruby (on Rails)

Ruby on Rails es un marco de desarrollo de aplicaciones web de código abierto bastante interesante. **Ruby** es el lenguaje de programación, **Rails** es la plataforma (o *framework*) que te permite construir aplicaciones web de una manera muy rápida y cómoda.

Los requerimientos en cuanto al entorno de ejecución de las aplicaciones son similares a los de las aplicaciones desarrolladas con *PHP*.

Generalmente, no está siendo tomado muy en serio por parte de las grandes empresas; pero debido a la velocidad de desarrollo que proporciona, muchas *startups* lo están utilizando en sus proyectos. Por tanto, el número de ofertas de empleo que podrás encontrar que requieran de estos conocimientos serán bastante limitadas.

En resumen, con *Ruby on Rails* podrás desarrollar fácilmente aplicaciones web (de una forma bastante rápida) que podrán ser ejecutadas sobre cualquier máquina (con cualquier sistema operativo) y utilizando herramientas de software libre.

Java

Java es un lenguaje de programación de propósito general. Con *Java* podrás desarrollar aplicaciones de escritorio, aplicaciones web e incluso aplicaciones móviles (*Android*).

Java se ejecuta en cualquier entorno (la plataforma está disponible para todos los sistemas operativos) y las aplicaciones web que desarrollemos tendrán que ser desplegadas en lo que sus creadores llaman un "*contenedor de servlets*" (como **Apache Tomcat**, disponible también para cualquier sistema operativo).

Existen diferentes entornos de desarrollo, aunque los más utilizados y extendidos son **Eclipse** y **Netbeans** (que están disponibles de manera gratuita). Normalmente también utilizaremos un sistema de bases de datos como **MySQL** o **PostgreSQL** (también disponibles de manera gratuita), aunque podríamos utilizar cualquier otro.

Al igual que *.NET*, *Java* es una plataforma de desarrollo que te permitirá trabajar para las grandes empresas, para las pequeñas empresas, para las empresas de productos web o para ti mismo (como *freelance* o emprendiendo tu propio proyecto). Además, hay muchas ofertas de empleo que requieren de estos conocimientos y se trata de una de las plataformas de desarrollo más populares del mundo. Por otro lado, la enorme expansión de los *smartphones* en los últimos años y del sistema *Android*, ha abierto un nuevo abanico de posibilidades muy interesante para los desarrolladores de *Java*.

En resumen, con *Java* podrás desarrollar aplicaciones web (y también aplicaciones de escritorio) que serán ejecutadas sobre cualquier máquina (con cualquier sistema operativo); así como aplicaciones móviles (*apps*) para *smartphones* o *tablets* con *Android*.

Objective-C

Objective-C es el lenguaje de programación que permite desarrollar **aplicaciones de escritorio** para los ordenadores *Mac* de *Apple* y **aplicaciones móviles (apps)** para los *smartphones* o *tablets* con el sistema *iOS* de *Apple* (*iPhone*, *iPod* y *iPad*).

Se trata de un lenguaje únicamente enfocado en desarrollar aplicaciones para entornos Apple, por lo que su uso está algo más restringido en este aspecto, aunque la rápida generalización y expansión de estos dispositivos en los últimos años lo convierten en una opción a tener en cuenta también.

Para desarrollar aplicaciones con *Objective-C* (recordemos, aplicaciones para *Apple*), necesitaremos utilizar un ordenador ***Mac*** y el entorno de desarrollo ***XCode*** (disponible de manera gratuita desde la *App Store de Mac*).

Personalmente, me parece una opción muy interesante (y el ecosistema de desarrollo de Apple es estupendo), pero nunca lo recomendaría como primer lenguaje de programación. Piensa además que te estarías restringiendo mucho en cuanto a tus posibilidades de trabajo, ya que sería prácticamente sólo apps y sólo para *iOS* (*iPhone* y *iPad*). Aún así, podrás encontrar bastantes ofertas de trabajo que requieren conocimientos de desarrollo de aplicaciones de este tipo, la mayoría provenientes de pequeñas *startups* o para trabajar como *freelance*.

En resumen, con *Objective-C* podrás desarrollar aplicaciones de escritorio para *Mac* y aplicaciones móviles (*apps*) para *iPhone/iPad*.

CAPÍTULO SIETE

¿POR DÓNDE EMPIEZO? EL MÉTODO DE 6 PASOS

¿Por dónde empiezo? El método de 6 pasos

A estas alturas ya sabes qué es y en qué consiste el desarrollo de aplicaciones y cuáles son los motivos por los que podrías querer convertirte en un desarrollador. Pero el problema es que quizás nunca hayas escrito ni una sola línea de código (o tal vez lo hiciste alguna vez, pero no has vuelto a hacerlo desde que estabas estudiando), así que ¿por dónde empezar?, ¿aprendo por mi cuenta o tengo que acudir a alguna escuela o universidad?

Soy un desarrollador autodidacta, pero también me licencié en Ingeniería Informática, por lo que conozco las dos caras de la moneda. Aprender a desarrollar aplicaciones a partir de un libro te enseña el lado práctico de cómo programarlas. El aprendizaje en la universidad te enseña la teoría de por qué lo hacemos de la forma en que lo hacemos.

Teniendo esto en mente, si yo estuviese empezando hoy de nuevo, seguiría los siguientes 6 pasos:

- 1) Aprender los fundamentos.
- 2) Desarrollar un pequeño proyecto.
- 3) Mejorar el aspecto visual de la aplicación.
- 4) Desplegar/distribuir la aplicación.
- 5) Ampliar los conocimientos.
- 6) Adquirir experiencia.

Paso 1: Aprende los Fundamentos

Lo primero que haría sería aprender los fundamentos necesarios para desarrollar aplicaciones. Esto pasa por aprender a programar (las bases de la programación, comunes a todos los lenguajes) y aprender los conceptos indispensables de bases de datos.

Lo haría del siguiente modo:

1. Escogería un lenguaje de programación (y sólo uno). Seguramente uno de propósito general, como **C#** (o **Java**).
2. Compraría un libro o haría un curso por Internet, que me enseñase a programar desde cero utilizando este lenguaje. Aquí aprendería los conceptos más básicos de la programación, las principales sentencias comunes a todos los lenguajes, los conceptos de la programación orientada a objetos, cómo crear un proyecto, cómo ejecutarlo y probarlo, etc.
3. Compraría otro libro u otro curso por Internet, que me enseñara los conceptos indispensables de bases de datos. Aquí aprendería a crear una base de datos y sus tablas, a definir la estructura de una tabla, a definir restricciones y relaciones, a hacer consultas a la base de datos, a insertar registros, modificarlos y eliminarlos. En este punto, me familiarizaría con el lenguaje **SQL** (lenguaje estándar utilizado para manipular bases de datos).
4. Escogería un lenguaje de programación web (y sólo uno). Seguramente **ASP.NET** (la opción perfecta para combinarlo con **C#**).
5. Compraría otro libro u otro curso por Internet, que me enseñase a programar aplicaciones web con este lenguaje. Aquí aprendería los conceptos de la programación web, me familiarizaría con la arquitectura *cliente-servidor*, con el mantenimiento de la sesión del usuario, etc.

Sé que podrías estar pensando: "menudo follón, ¿todo esto es necesario para dedicarse al desarrollo?". Piensa una cosa: desarrollar aplicaciones es una profesión; una profesión con el potencial suficiente para liberarte y permitirte cambiar tu vida. ¿Realmente piensas que es tanto aprender estas tres cosas a cambio de todo lo que te ofrece? ¿Cuántas cosas distintas crees que tiene que aprender un arquitecto, o un médico, o un abogado...?

Te aseguro que no es tanto como parece, ni tan complicado (no necesitas saberlo todo... ¡ni mucho menos! tan sólo lo básico de cada cosa). Pero lo que sí vas a necesitar es ponerte en buenas manos. No es fácil encontrar a alguien que sepa transmitir estos conocimientos tan técnicos a personas que se están iniciando (piensa que son conceptos que probablemente no te sonarán de nada, que quizás nunca antes lo has visto...).

Paso 2: Desarrolla un pequeño proyecto

Una vez conociera los fundamentos más básicos, lo siguiente que haría sería escoger un pequeño proyecto y... ¡desarrollarlo!

Serviría cualquier proyecto que nos resolviera un problema concreto. Por ejemplo: una aplicación para gestionar tus gastos mensuales, una aplicación para llevar un control de las tareas que tienes que hacer cada día o una base de datos de nuestras recetas preferidas... cualquier cosa que se te ocurra y que te resuelva un problema concreto.

Para ello, descargaría e instalaría un entorno de desarrollo integrado como, por ejemplo, **Microsoft Visual Studio** (la versión *Express* es gratuita). Esta herramienta nos permitirá editar y organizar los ficheros de nuestro proyecto, así como ejecutarlo para que podamos probarlo.

Además, descagaría e instalaría un sistema de bases de datos como, por ejemplo, **Microsoft SQL Server** (la versión *Express* es gratuita). Estos sistemas incluyen una herramienta para administrar las bases de datos con la que podremos crear y definir la base de datos que almacenará los datos de nuestro proyecto, y que será accedida desde la aplicación que desarrollemos.

En este momento no me preocuparía por el aspecto visual de la aplicación, sino simplemente por que funcione correctamente.

Paso 3: Mejora el aspecto visual de la aplicación

Una vez tuviese la aplicación funcionando correctamente, me centraría en mejorar su aspecto visual, para que resultase más atractiva a los ojos de los demás.

Para ello, primero compraría un libro o haría un curso por Internet sobre diseño web. Aquí aprendería los fundamentos del diseño de interfaces de usuario, las hojas de estilo, la maquetación y organización visual de la aplicación, etc.

Después, aplicaría estos conceptos al proyecto desarrollado anteriormente. Cambiaría los controles estándar de la aplicación (botones, campos para introducir texto, etc.) por controles más bonitos y que diesen un aspecto más profesional a mi aplicación. Organizaría bien los distintos módulos de mi aplicación y los pondría accesibles desde una bonita barra de menú, etc., etc.

Paso 4: Despliega/distribuye la aplicación

En este punto, ya tendría una aplicación que resuelve un problema concreto, que funciona y además está chulísima visualmente. El único problema es que estaría en mi ordenador y sólo yo podría apreciarla. ¡Una pena!

Por tanto, lo que haría sería ponerla disponible en Internet (posiblemente, en mi aplicación habría implementado un mecanismo de acceso basado en usuario-contraseña para que mis datos no estuviesen a la vista de cualquiera). Para hacer esto, abriría una cuenta de **hosting** con algún proveedor de alojamiento en Internet. Estas empresas lo que hacen es que te alquilan un espacio en un servidor de Internet para que puedas alojar allí tu aplicación. Esto tiene un coste mensual, pero hay empresas con tarifas muy competitivas (e incluso puedes encontrar algunas que ofrecen versiones gratuitas con espacio limitado pero que pudieran ser útiles como punto de partida si no quisiésemos afrontar este pequeño gasto desde un primer momento).

Una vez tuviese la cuenta de *hosting* abierta, subiría mi aplicación para alojarla en el espacio que me hubiesen asignado y... *voilà*, ¡ya tenemos nuestra aplicación en Internet!!

Paso 5: Amplía tus conocimientos

Después de desarrollar este primer proyecto y desplegarlo en Internet, probablemente lo siguiente que haría sería profundizar más y ampliar mis conocimientos.

Intentaría averiguar en qué consiste el desarrollo corporativo, es decir, de qué forma desarrollan las empresas las aplicaciones del "mundo real". Me formaría y aprendería las principales técnicas que utilizan para abordar los proyectos, cómo estructuran el código en "capas" para que sea reutilizable, cómo implementan cada una de estas capas, etc.

Por otro lado, también investigaría sobre el mundo de las aplicaciones móviles. Escogería una de las principales plataformas (*Android* o *iOS*) y me formaría y aprendería cómo desarrollar este tipo de aplicaciones, cómo distribuirlas a través de las tiendas de apps, etc.

Muy probablemente, en este paso tuviese que aprender algún lenguaje de programación más y utilizar alguna herramienta adicional. Como ya hemos dicho antes, un buen desarrollador tiene que estar siempre dispuesto a aprender cosas nuevas para estar al día.

Paso 6: Adquiere experiencia

Ahora que ya sabría lo que es desarrollar aplicaciones (incluso tendría desarrollada ya una) y habría profundizado en aspectos un poco más avanzados y explorado nuevas tecnologías, lo siguiente que haría sería adquirir experiencia desarrollando aplicaciones (aprender haciendo).

Esto podría hacerlo de dos maneras:

A) Desarrollando otros proyectos por mi cuenta.

Un punto curioso que tiene el desarrollo de aplicaciones con respecto a otras disciplinas de la ingeniería es el hecho de que hay personas que lo hacen como un hobby. Nadie (o casi nadie) diseña sistemas mecánicos por diversión, pero miles de personas desarrollan aplicaciones en su tiempo libre simplemente por el desafío que supone y la satisfacción que obtienen al crear algo. Construir algo por tu cuenta no es tan complicado como parece. Con un poco de programación y el conocimiento aprendido en los pasos anteriores, podríamos tener una aplicación web sencilla construida en cuestión de días.

Cada pequeña porción de experiencia que adquiramos de esta manera, será fundamental para conseguir un trabajo como desarrollador. Conseguir un trabajo sin tener demasiada experiencia en desarrollo profesional no es algo tan complicado. Conseguir un trabajo sin absolutamente ninguna experiencia, sí que lo es.

Aunque si bien es cierto que tener experiencia en desarrollo profesional es lo ideal y que quizás algunas ofertas de empleo requieran de algunos años, no es menos cierto que si no la tenemos, en algún momento tenemos que comenzar a adquirirla. Si queremos optar a un puesto como desarrollador de aplicaciones, y hemos construido al menos un proyecto de ejemplo y lo hemos puesto accesible desde Internet, estaremos en una posición mejor que el resto de candidatos al puesto.

B) Participando en proyectos de Código Abierto.

Según *Wikipedia*:

El término se utilizó por primera vez en 1990 las comunidades de software libre, tratando de usarlo como reemplazo al ambiguo nombre original en inglés del software libre (free software). «Free» en inglés significa dos cosas distintas dependiendo del contexto: gratuidad y libertad. Lo cual implica, para el caso que nos ocupa, "software que podemos usar, escribir, modificar y redistribuir gratuitamente" (software gratuito) y, además, software libre, según la acepción española de libertad.

Esto significa que cualquier persona, incluso un nuevo desarrollador, puede contribuir a un proyecto de código abierto. Es posible que hayas oído hablar sobre *Linux*, *Apache*, *Wordpress*, *PHP* y *Java*; todos son proyectos de código abierto.

Puedes ir a <http://sourceforge.net/> o <http://www.codeplex.com>, y echar un vistazo. Puede resultar algo intimidante al principio pero hay un montón de proyectos que podrían aprovechar tu ayuda. Escoge un proyecto que sea de tu interés y lee las condiciones sobre cómo puedes involucrarte en él. Es posible que tengas que solicitar el acceso. En cualquier caso, la incorporación de "*Open Source Contributor*" como mérito en tu currículum será una gran ventaja sobre el resto.

CAPÍTULO OCHO

ALGUNAS CONSIDERACIONES FINALES

Algunas consideraciones finales

El método en 6 pasos que acabo de compartir contigo, constituye tan solo uno de los múltiples caminos que se podrían seguir para convertirse en un desarrollador de aplicaciones y ganarte la vida con ello, ya sea consiguiendo que te contrate una empresa, trabajando como *freelance* o emprendiendo un proyecto propio.

Hay una cosa que me gustaría aclararte. Soy consciente de que en los primeros pasos del método hice referencia a los lenguajes de programación y herramientas de **Microsoft**. Alguno se estará preguntando, ¿por qué? Pues lo hice sencillamente porque es la recomendación personal que le haría a un amigo (y a ti te considero ya uno de ellos) que estuviera empezando en el mundo del desarrollo de aplicaciones. El "ecosistema" de *Microsoft (.NET)* tiene, a mi modo de ver, grandes ventajas:

- ❑ La primera es que casi todo el mundo tiene a su disposición un ordenador con sistema operativo *Windows*.
- ❑ La segunda es que las herramientas son estupendas y están disponibles de forma gratuita (en su versión *Express*).
- ❑ La tercera es que *C#* es un lenguaje de propósito general (nos servirá para aplicaciones web, aplicaciones de escritorio, aplicaciones móviles...), además su sintaxis es muy parecida a la de otros lenguajes como *Java*, y lo considero apropiado como primer lenguaje para aprender.
- ❑ Y por último, la plataforma *.NET* tiene una gran demanda en el mercado.

En cualquier caso, si prefieres empezar a desarrollar con cualquier otra tecnología, como por ejemplo *PHP*, los pasos serían los mismos y sólo tendrías que adaptar los dos primeros (aprender los fundamentos del lenguaje *PHP* y familiarizarte con la base de datos *MySQL*, y desarrollar tu primer proyecto con ellos).

El método que hemos compartido nos aporta una serie de ventajas interesantes:

1. Te orienta a la acción de desarrollar una aplicación. No hay nada más inútil que tirarse 6 meses leyendo libros sobre programación. Tienes que empezar a desarrollar tan pronto como sea posible.
2. Aprenderás los fundamentos y sentarás las bases utilizando desde el comienzo un lenguaje comercial. Las aplicaciones reales se construyen utilizando lenguajes de programación como *ASP.NET* o *PHP*. De este modo, vas a aprender un lenguaje de programación con en el que probablemente conseguirás un trabajo en un par de meses.
3. Crearás un proyecto para tu portfolio. Está claro que no va a ser la cosa más atractiva que hayas visto jamás, pero será una aplicación que podrás mostrar a un posible empleador, y servirá como prueba de que eres capaz de desarrollar aplicaciones.
4. Además, al terminar el paso 2 ya habremos averiguado si de verdad nos gusta esto del desarrollo de aplicaciones. En este punto del proceso, ya habremos experimentado algunos de los problemas que podremos encontrarnos al dedicarnos a esto, y nos haremos una idea del tiempo y el esfuerzo que supone desarrollar una aplicación.

Por mi parte nada más. Ha sido un auténtico placer compartir esta información contigo, y espero y deseo que te sea de utilidad.

Recuerda que podrás contactarme siempre que lo necesites, a través del formulario de contacto de mi blog (escueladeinformaticos.com), o escribiendo directamente un email a: info@escueladeinformaticos.com

Te deseo muchos éxitos en tu nueva etapa como desarrollador!

Un fuerte abrazo,

Julio Liarte

ESCUELA DE INFORMÁTICOS

Entrenándote para que puedas vivir de la informática

¿Necesitas ayuda?

¿No sabes muy bien cuál sería la mejor manera de empezar?...

Visita mi web escueladeinformaticos.com y descubre cómo puedo ayudarte a que recorramos estos pasos juntos.



Acceder



escueladeinformaticos.com