

# **cómo ser un micro-isv**

# cómo ser un micro-isv

- **Introducción**
- **Definición del *target***
- **Desarrollo del producto**
- **Comercialización**
- **Soporte**
- **Bibliografía y enlaces recomendados**

# introducción

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

## ¿ Qué es un isv ?

Motivación  
Shrinkwrapped vs consultingware  
Fundamentos del éxito  
Problemas habituales  
Modelo de empresa

- **ISV – *independant software vendor***
- **Microempresa o trabajador independiente que vende software empaquetado por internet**
- **Continuación del desarrollo shareware**

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

¿ Qué es un isv ?

## Motivación

Shrinkwrapped vs consultingware  
Fundamentos del éxito  
Problemas habituales  
Modelo de empresa

- **Realización personal**
- **Ser tu propio jefe, tomar tus decisiones**
- **Aprender facetas externas a la programación**
- **Crear nuevas expectativas profesionales**
- **Aspecto económico**
- **Facilidad de puesta en marcha**
- **Sin ataduras**
- **Libertad de horarios**

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

¿ Qué es un isv ?

Motivación

**Shrinkwrapped vs consultingware**

Fundamentos del éxito

Problemas habituales

Modelo de empresa

- **Shrinkwrapped:**

Software *empaquetado*

Software listo para usar

Software dirigido a colectivos

Precio asequible, menos de 50\$

NO es desarrollo a medida ni consultingware

La idea es vender muchas licencias de 1 sólo producto

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

¿ Qué es un isv ?

Motivación  
Shrinkwrapped vs consultingware  
**Fundamentos del éxito**  
Problemas habituales  
Modelo de empresa

- **el programa debe responder a las necesidades reales de los posibles usuarios**
- **el programa debe ser realmente bueno, y no algo de segunda fila**
- **los potenciales usuarios deber tener conocimiento de que el programa existe**
- **el programa debe llegar a estos usuarios**
- **tiene que haber una razón para que paguen por él**

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

¿ Qué es un isv ?

Motivación  
Shrinkwrapped vs consultingware  
Fundamentos del éxito  
**Problemas habituales**  
Modelo de empresa

- **Ser un ISV no es sólo programar**
- **Te tienes que ocupar de todo**
- **Problemas de falta de disponibilidad de tiempo**
- **Problemas de falta de decisión**
- **Problemas de falta de constancia**
- **Falta de resultados a corto plazo**

## Introducción

Definición del target  
Desarrollo del producto  
Comercialización  
Soporte  
Bibliografía y enlaces

¿ Qué es un isv ?

Motivación  
Shrinkwrapped vs consultingware  
Fundamentos del éxito  
Problemas habituales  
[Modelo de empresa](#)

- **Fog Creek**      <http://www.fogcreek.com>
- **Bradbury**      <http://www.bradsoft.com>
- **Tune-Up**      <http://www.tune-up.com>
- **GoodSol**      <http://www.goodsol.com>
- **Essential PIM**      <http://www.essentialpim.com>

# definición del target

- **Qué tipo de software vamos a desarrollar**
- **Realizar especificaciones funcionales**
- **Qué precio va a tener**
- **Recomendaciones**

Resolver problemas específicos, pero generalistas

Evitar software para *geeks*

- **Modo de licenciar el producto**

{ software propietario, mixto, freeware, GPL, BSD }

- **Versiones a realizar**

{ registrada, evaluación, gratuita }

- **Manera de activación**

{ sin activación, activación por clave, ligado a máquina }

- **Estudiar productos similares**

Fortalezas y debilidades con los competidores

Precio

- **Estudiar alternativas libres o gratuitas**

- **Valorar si existe un nicho por cubrir y la manera de atacarlo**

# desarrollo del producto

- **Elegir la herramienta-entorno adecuada**  
evitar runtimes e instalaciones farragosas
- **Contar con betatesters independientes**
- **Liberar el programa unicamente cuando esté terminado**
- ***“Un buen software cuesta 10 años”***

- ***“El software grandioso requiere una devoción fanática por la belleza”***
- **Seguir los estándares de la plataforma**
- **Usar el interfaz como elemento diferenciador**
- **Usar elementos gráficos con aspecto profesional**
- **Conseguir un estilo propio**

- **Definir qué tipo de documentación acompaña al programa**

{ fichero hlp | chm | doc | pdf, ayuda en linea }

- **La documentación debe:**

explicar qué hace el programa

explicar donde obtener ayuda adicional

incentivar y facilitar la compra del programa

- **Paquete autocontenido**  
{ software, documentación, datos }
- **Crear grupo de programas y link en escritorio**
- **Probar cada vez la instalación tanto en limpio como actualizando versiones**
- **Proveer utilidad de desinstalación**

# comercialización

- **Decidir cual va a ser el canal de ventas**  
{ sitio web propio, sitio web ajeno, publicaciones o catálogos }
- **Decidir la manera de enviar el producto**  
{ paquetería, ftp privado, clave por e-mail }
- **Preparar la infraestructura y logística \*antes\* de lanzar el producto**

- **Claro, concreto y conciso**
- **Escaparate y sitio de ventas**
- **Permitir descarga de demo/evaluación/gratuito**
- **El proceso de compra debe estar claro**
- **Cobro por tarjeta, transferencia, paypal,...**
- **Aportar valor añadido**
- **Elementos de promoción:**  
{ Listas de correo desatendidas, blog corporativo, foros de soporte }

- **Correos a sitios web de software**
- **Notas de prensa a publicaciones escritas**
- **Lista de correo del sitio web**
- **Promoción en sitios web relacionados**

- **Datos personales de usuarios**
- **Control de licencias**
- **Acceso a actualizaciones**
- **Direcciones de e-mail para lista de correo**
- **Seguimiento de ventas**
- **Listados fiscales**

# sopORTE

- **Definir política de devolución de pago**
- **Incluir en el sitio web formulario de contacto, enlace de e-mail y foros de soporte**
- **Definir el tipo de soporte a realizar**  
{ e-mail, foros, teléfono, messenger,... }
- **Diferenciar entre soporte y consultorio privado**
- **Relacionarse con los usuarios para conseguir betatesters de los programas**

- **Contestar siempre todos los correos o post**
- **Contestar en un plazo breve – máximo 24 horas**
- **Cuando no se pueda pedir disculpas**
- **Ser amable con el usuario y no entrar en debates**
- **Tener contestaciones predefinidas**

# **bibliografía y enlaces**

- **The pragmatic programmer**
- **User interface design for programmers**
- **About face**
- **Presos de la tecnología**
- **El ordenador invisible**
- **Diseño de sistemas interactivos**

- **Las 22 leyes inmutables del marketing**
- **Liberando los ideavirus**
- **La frontera del éxito**

- **Joel on Software** · <http://www.joelonsoftware.com>
- **Eric Sink** · <http://software.ericSink.com>
- **Paul Graham** · <http://www.paulgraham.com>
- **A shareware life** · <http://www.asharewarelife.com>
- **Todo Or Else** · <http://www.todoorelse.com>
- **Soft in Spain** · <http://www.softinspain.com>
- **avemundi** · <http://www.avemundi.com>