**Traducción del Contenido del video CQ4  
Lucas Salamanca**

(0:00) Acabas de empezar a aprender Python y quieres desarrollar un proyecto genial después de aprender los (0:06) conceptos básicos. Bueno, esta serie de videos es exactamente para ti porque en esta serie (0:11) vamos a desarrollar un juego con Python. Ahora bien, si conoces los conceptos básicos y tomaste algún (0:17) curso básico, sin importar el canal o el curso que sea, entonces esta es la serie que (0:23) estás buscando porque te dará los fundamentos de cómo empezar a estructurar un proyecto de Python. (0:29) Ahora bien, en esta serie, desarrollaremos un juego llamado Buscaminas, que se conoce como un juego para un solo (0:36) jugador que es muy divertido de jugar y también desafiante. Veamos qué juego aprenderás (0:41) a desarrollar en esta serie. Bien, ese será el juego que desarrollaremos (0:46) a lo largo de la serie.

Ahora pueden ver que tenemos varias celdas en las que podemos hacer clic (0:52) para abrirlas. El objetivo principal del buscaminas es no hacer clic en una celda que tenga una mina detrás (0:59). Esto significa que si la abro, recibo (1:04) un número.

Esto significa que en las celdas que rodean esa celda, hay una mina en la que (1:11) no debo hacer clic. Así que voy a suponer que no hay ninguna mina aquí. Y tampoco (1:17) hay ninguna mina allí.

Y pueden ver que tengo razón. Así que haré clic derecho (1:23) para marcarla como mina. Porque si hago clic izquierdo ahí (1:29), perderé la partida.

Ahora intentaré adivinar las ubicaciones restantes (1:35) que no tienen mina. Haré clic ahí. Verán que (1:40) recibo algunos números; por ejemplo, alrededor de la celda, no tengo ninguna mina.

Así que es seguro (1:47) hacer clic aquí, aquí y allá. Ahora intentemos hacer clic izquierdo en esa ubicación. Verán (1:54) que dice que hicieron clic en una mina y perdieron la partida.

El objetivo principal es adivinar (1:59) todas las ubicaciones que no tienen mina y así ganarán la partida. Así que (2:06) tendremos que pensar mucho para desarrollar el juego. Y va a ser genial (2:11) lidiar con todos los desafíos que este juego conlleva.

Así que, antes de empezar, (2:16) agradecería que le dieran "Me gusta" a ese video; esto ayudará a difundirlo en (2:20) todo YouTube. Y si conocen a alguien que esté empezando a aprender Python y (2:25) quiera desarrollar un proyecto interesante, invítenlos a ver mi serie. Podría ser muy (2:31) útil para aquellos videos que están en el nivel intermedio entre principiantes y expertos (2:36).

Dicho esto, comencemos. Antes de empezar a escribir este proyecto, (2:50) espero que tengan Python instalado en su computadora y un ID listo (2:57) para usar y que reconozca el intérprete de Python en su computadora. Usaré Python 3.8. (3:03) Pero mientras tengas una versión superior, no deberías tener problemas para escribir este (3:09) proyecto.

Ahora bien, no usaremos demasiadas bibliotecas externas ni demasiado código que (3:15) dependa de la versión de Python. Así que no hay problema si incluso usas Python 3.10 o Python 3.11 (3:22), que probablemente se lanzará en el futuro. Bien, comencemos.

Ahora, para escribir (3:28) un juego de mesa, un juego 2D, podemos usar muchas bibliotecas disponibles en Python. (3:34) En este, usaremos TK inter, que es muy útil y una excelente biblioteca para practicar (3:40) porque incluye muchas clases útiles que instanciaremos para crear nuestra (3:47) ventana. Comenzaremos importándolo de la siguiente manera. Diré desde (3:52) que importen todo. Esto será un excelente punto de partida. La siguiente línea que escribiré (3:59) es para instanciar una instancia de ventana.

Probablemente queramos hacer esto porque nuestro proyecto (4:05) estará en una ventana. Llamaré a mi variable "root", que será igual a (4:11) una instancia de la palabra clave TK. Este es el elemento más básico que se puede crear.

(4:18) Esta es una ventana normal. Al ejecutarla, no ocurrirá nada porque (4:23) necesitamos indicarle a TK que se ejecute hasta que la cerremos con el botón X en la esquina superior derecha (4:30). Para ello, diremos "root dot main loop" así, así, así que necesitamos llamar a ese método (4:36) y todo el código estará entre esas dos líneas. Si lo ejecutamos (4:41), como pueden ver, tenemos la ventana más básica posible. Ahora, una breve acotación: (4:47) la razón por la que usé la variable raíz es simplemente una convención en los interproyectos de TK. Así que... Te resultará más fácil (4:53) buscar problemas en Stack Overflow, por ejemplo, si sigues estas (4:57) convenciones.

Bien, veamos algunos atributos que podemos cambiar aquí (5:01) para mejorar nuestra ventana. El primero, obviamente, será cambiar el tamaño de la (5:07) ventana. Podemos hacerlo llamando a la función de geometría, es decir, al método de esta ruta.

Y esto (5:14) acepta una cadena similar a la siguiente: peso, x, altura. Y aquí, obviamente (5:22) puedes poner los números que quieras. Ahora, presta atención a lo que sucedería si los reemplazara (5:27) con números reales como los siguientes: verás la diferencia inmediatamente (5:34). Y si lo ejecutara, podrás ver cómo cambia. Ahora también podemos prestar atención a que (5:39) usa esta cadena como título de nuestra ventana. Podemos cambiarlo llamando a un método llamado (5:46) "title" y asignarle el título que queramos. Así, podemos jugar a un buscaminas (5:52) y volver a ejecutar nuestro programa. Y pueden ver los cambios.

No mencioné esto. Pero por defecto (5:58), todas las ventanas son redimensionables. Y eso puede ser molesto al crear muchos (6:04) elementos dentro de esta ventana, como botones o marcos.

Por eso probablemente buscamos evitar (6:12) redimensionar esta ventana, ya que nos facilitará la tarea al añadir más (6:17) elementos. Esto se puede lograr llamando al método redimensionable (6:25) de esa manera, y básicamente especificando que no se nos permite redimensionar el ancho (6:30) ni la altura de esta ventana. Esto se logra simplemente pasando "false" dos veces: (6:38) una para el peso y otra para la altura.

Si volviera a ejecutar nuestro programa (6:43), verían el efecto: esta función de maximizar está deshabilitada. Esto nos facilitará el desarrollo del juego (6:49) en el futuro. (6:54) Después de llegar a esa etapa, quizá también nos interese cómo cambiar el color de fondo (6:59) de nuestra ventana.

Podrían usar valores hexadecimales si saben cómo usarlos (7:05). Básicamente, admite muchas palabras clave para especificar los colores. No queremos (7:11) ocuparnos demasiado del estilo; nos centraremos en la lógica del algoritmo para (7:17) escribir un juego de este tipo.

Para ello, lo simplificaré. Aun así, queremos (7:23) tener una ventana bien visualizada. Así que voy a la quinta línea.

Y (7:28) usaré root dot configure. Y necesitaremos configurar el color de fondo para nuestra (7:34) ventana. Así que necesitamos llamar a este método y pasar BG igual al color (7:41) que queramos.

Ahora, en la descripción, pueden encontrar una lista de colores disponibles usando (7:48) colores con la palabra clave. En la descripción, hay un enlace con una larga lista de colores (7:54) disponibles. Voy a usar negro para esta ventana. Y eso es todo.

Así que pueden ver el (8:00) efecto inmediato de cambiar el color de fondo. Bien, ahora que entendemos esto (8:05), veamos cómo podemos crear algunos elementos dentro de nuestra ventana. Ahora vamos a (8:11) comenzar con el elemento más básico que se puede crear en una ventana, llamado marco.

(8:17) Un marco es como un contenedor que podría contener más elementos que crearemos en el futuro (8:24). Queremos usar esos marcos porque, obviamente, queremos dividir nuestra ventana en varias (8:31) secciones. Así será más fácil entender dónde queremos ubicar cada uno de nuestros botones o (8:37) textos en el futuro.

Por ejemplo, podríamos dedicar esa área de nuestra ventana a un título (8:44) a la derecha, así que tendríamos el título aquí. Y podríamos decidir que nos gustaría tener una barra lateral (8:49). Así que podríamos dedicar el área izquierda de nuestra ventana a la barra lateral para mostrar el marcador (8:57), por ejemplo, algo así.

Y podríamos tener otro marco aquí. Así podríamos mostrar (9:03) el partido aquí. Es una gran idea dividir la ventana en varios marcos. Será útil visualizar diferentes elementos más adelante.

Ahora (9:14) escribiré un comentario que diga "anular la configuración de la ventana". Justo antes del bucle principal (9:25), también comentaré "ejecutar la ventana". Aquí está la ubicación exacta (9:32) donde comenzaremos a crear algunos marcos.

Empecemos. Dijimos (9:37) que queríamos un marco ubicado en la parte superior. Llamaré a esta variable "marco superior" (9:43), que corresponderá a una clase llamada marco.

Una vez que instanciamos esta clase de marco (9:51), esta será responsable de instanciar un marco, lo cual decidiremos más adelante (9: 57) sobre dónde queremos ubicarlo en nuestra ventana. El primer argumento que debe recibir este marco (10:03) es qué elemento queremos ubicar. Se ubicará en la ventana (10:11).

Así que pasaremos `route`. Y luego los separaré en varias líneas. Así (10:17) será más fácil de leer.

Como pueden ver, los paréntesis se abren aquí y se cierran aquí. (10:23) Ahora, vamos a darle más atributos. Primero, queremos darle un fondo diferente (10:29) para poder diferenciar bien entre el marco y la ventana, ya que será (10:34) difícil entender dónde se encuentra el marco si mantenemos el mismo color.

Por razones de depuración (10:39), mantendré este cambio de rojo a negro más adelante, y lo comentaremos aquí. Y luego (10:46) aquí abajo, decidiré el ancho y la altura. Diré que el ancho debe ser igual a (10:53) el mismo tamaño de ancho que le dimos a la ventana, ya que queremos cubrirla por completo.

Entonces (11:00) este debería ser el ancho completo de la ventana y la altura debería ser aproximadamente 180. (11:08) Así que es simplemente 720 dividido entre cuatro, y eso tiene mucho sentido. Y ahora que tenemos esto, (11:15) necesitamos decidir dónde queremos comenzar este marco.

Para hacer esto, (11:20) voy a decir guion bajo superior marco punto lugar. Ahora, este lugar recibirá (11:28) dos argumentos que son obligatorios. Y antes de pasarlos, permítanme mostrarles cómo funciona el método (11:33) lugar en la ventana. Entonces, cuando usamos el método de ubicación, necesita recibir (11:39) el valor en píxeles. Sabemos que tenemos un fotograma que queremos que comience exactamente desde aquí. Y (11:49) sabemos que probablemente debería cubrir esta área, ¿verdad? Pero ¿cómo le indicamos que comience (11:55) desde aquí? Bueno, en la ubicación, necesitamos especificar los ejes x e y, lo que significa que si queremos (12:05) comenzar desde la esquina superior izquierda, entonces necesitamos darle el valor 0.0. Porque ahí es donde todo (12:14) comienza en términos de píxeles.

Ahora, el área completa aquí es de 1440. Porque ese es el eje x y el eje (12:27) y es de 720 píxeles. Entonces, la altura completa aquí es de 720.

Ahora bien, si, por ejemplo, quisiéramos tener un botón (12:40), solo por ejemplo, no lo haré en esta ocasión. Digamos que queremos un botón (12:45) que comience aquí. Lo colocamos en el centro de la página (12:51), ¿verdad? Necesitamos dividir ese valor entre dos.

El eje x es 720. Sabemos (13:01) que queremos que esté en la parte superior tanto como sea posible. Este debería ser el valor (13:08) del eje y, que será cero.

Así funciona el método "place". Necesitamos especificar el valor en píxeles (13:16) para los ejes x e y. Limpiaré todo lo que dibujé aquí y volveré a (13:24) pi charm para indicar los valores de x e y. Ahora vean cómo se visualizará este fotograma (13:33).

Si detengo el proceso y lo vuelvo a ejecutar, verán que este es el resultado esperado (13:39), tal como queríamos. Empezó desde aquí y tomó toda el área de peso. Y tomó (13:48) 180 píxeles de la altura.

Eso es exactamente lo que queríamos. Ahora, solo para asegurarnos de que (13:55) lo entendieron, voy a cambiar esto de x a 20. Y adivinarán qué sucederá.

Entonces (14:01) si volvemos a ejecutarlo, obviamente comenzará desde una posición ligeramente a la izquierda, (14:06) pueden ver dónde está ubicado, ¿verdad? Empieza justo aquí. Y si saltamos la y por 20, (14:14) y nos detenemos y volvemos a ejecutar esto, entonces pueden ver el resultado. Así es como funciona el método de lugar.

(14:20) Obviamente, voy a volver a cambiarlos a cero y continuar desde aquí. Bien, ahora que (14:27) lo entendimos, sigamos adelante e intentemos crear un fotograma más. Ahora, esta vez, quiero (14:32) crear un fotograma que usaremos como barra lateral izquierda, donde mostraremos quizás la puntuación o algo (14:38) útil.

Así que voy a crear una variable de fotograma izquierdo. Esta será (14:43) igual al fotograma. Instanciaremos una más de esa clase.

Y queremos este fotograma en (14:50) la fruta. Y luego, si decimos algo como "Fondo" es igual a otro color, simplemente copiaré esa línea (14:57) y la pegaré. Y usaremos algo como azul.

Así, podemos separar los diferentes fotogramas (15:02). El ancho será 360. Estoy usando intencionalmente un valor que es solo una cuarta parte de ese valor (15:13).

Así me será más fácil entender qué porcentaje del ancho total cubrimos (15:20) con mis fotogramas. Ahora que especifiqué el ancho, voy a especificar la altura. Así que (15:26) diré que estos deben cubrir toda la altura de nuestra ventana.

Ahora, presten atención a que (15:33) ya cubrimos 180 píxeles para el marco superior. Así que no podemos decir simplemente 780, sino 720 así. Necesitamos avanzar (15:42) y recordar la altura de toda la ventana y restar este número por ese número (15:51), así que deberíamos estar pasando el año 540. No se preocupen si creen que codificamos demasiados valores (15:59), me aseguraré de no tener ningún valor codificado más adelante (16:04) cuando reestructuremos este proyecto en el futuro.

Ahora nos centramos en comprender (16:09) cómo interactúa TK. Ahora que lo hemos entendido, digamos que el fotograma izquierdo (16:17) debe colocarse en x igual a cero, pero y debe ser igual a 180. Porque queremos (16:26) comenzar justo después de donde se encuentra el fotograma superior.

Y si ejecutamos nuestro programa (16:34), pueden ver que tenemos el resultado perfecto. Para que lo entiendan bien, (16:38) si tomamos el eje y y lo saltamos 20 píxeles, pueden ver que comienza justo aquí (16:47) porque saltó 20 píxeles en el eje y. Por eso quiero asegurarme de mantener el valor (16:56) de 180.

Bien, ahora que hemos llegado a esta etapa, probablemente noten que (17:03) tendremos demasiados números codificados en términos del peso, la altura y los (17:09) píxeles, etc. Por eso es una buena idea tener un archivo separado para todas nuestras constantes y (17:16) tener al menos algunas variables almacenadas en algún lugar. No voy a incluir eso en nuestro archivo main.py (17:23) porque podría ser demasiada información para almacenar en un solo archivo.

Además, (17:29) voy a obtener un nuevo archivo de Python. Y lo llamaré "configuración". Aquí dentro, puedo usar (17:35) algunas constantes útiles.

Y para mostrarles las ventanas una al lado de la otra, permítanme usar (17:41) la opción de clic derecho y dividir verticalmente. Así podemos empezar a pensar en algunas buenas variables (17:47) que nos gustaría almacenar aquí. Por ejemplo, podemos empezar con el ancho de nuestra ventana (17:53); podemos decir que el ancho es igual a 1440.

Podemos decir que la altura es igual a 720. Y (18:06) podemos volver inmediatamente a nuestro archivo main.py y usar algo como "importar configuración". Y luego (18:14) podríamos usar una cadena formateada.

Y podríamos convertirlas en esa cadena (18:23), podemos decir aquí "configuración ancho". Y podemos usar "configuración altura". Y (18:35) ese es un gran cambio en nuestro proyecto.

Ahora, si ejecuto nuestro archivo principal, pueden ver (18:41) que nada ha cambiado realmente. Y estamos en buena posición para continuar realizando esas manipulaciones (18:47) para almacenar más variables. Si recuerdan, dije en el tutorial que usé intencionalmente (18:54) esos números.

Así podríamos tener un porcentaje preciso del ancho. Por ejemplo, (19:01) había una gran razón para usar el valor 180 en la altura de los marcos superiores, porque 180 por cuatro (19:10) es 720. Así que puedo decir que me gusta consumir el 25% de la altura total de nuestra ventana.

Y esos (19:19) son aspectos importantes, porque no tendremos demasiados valores predefinidos. Así que lo que podemos hacer (19:24) ahora es tener un archivo para algunas funciones que podrían ser útiles para calcular esos datos. Por ejemplo, podríamos tener una función que calcule el 20% de la altura total de nuestra ventana (19:38).

Podemos usar otro archivo de Python llamado (19:44) utilidades. También podríamos tener una función que calcule el porcentaje de nuestra altura (19:52). Primero, podríamos importar la configuración.

Importaremos la configuración (20:00) de nuestro proyecto. Podemos usar una función. Llamemos a esta función "altura" (20:06) "guión bajo PR CT", una versión abreviada de "porcentaje".

Podemos recibir (20:12) el porcentaje como parámetro, haciéndolo un poco más pequeño. Luego, (20:19) podemos devolver el valor basándose en el argumento recibido. Podemos usar "return" (20:26). Puedo decir que la altura del punto de configuración se divide entre 100. Y puedo (20:35) cubrir esto con paréntesis. Y puedo multiplicarlo por el porcentaje.

Ahora quiero validar que he hecho un buen trabajo escribiendo esta función. No quiero llamar a mis funciones inmediatamente (20:47), sino simplemente intentar usarlas, ya que podría tener un error al escribir una función (20:53). Y eso también es una gran ventaja al ubicar diferentes funciones en archivos diferentes (20:58), ya que te dará una gran ventaja al depurar tus cosas. Por ejemplo, podría (21:03) ejecutar temporalmente solo el archivo utils.punto.py y probar mi función. Podría (21:09) intentar llamarla y pasar el año 25 y validar que realmente recibiré 180 de vuelta.

Así que puedo (21:17) usar print y ejecutar utils. Así que no estoy ejecutando mi proyecto original. Y ahora, si avanzas (21:27) y ves el valor aquí, Puedo ver que realmente recibo 180.

Y eso es bueno. Significa (21:33) que hemos hecho un gran trabajo al escribir esta función. Y ahora puedo empezar a usarla (21:39) en nuestro proyecto original, es decir, en el archivo main.py.

Así que puedo importar también todo el archivo utils (21:47), puedo decir "import utils". Y puedo llamar a la función de porcentaje de altura (21:54). Puedo usar utils dot height prct y pasar 25. Y, de nuevo, esto no tendrá ningún efecto.

Esto significa (22:03) que hemos hecho un gran trabajo. Solo que ahora somos más dinámicos y no codificamos valores de forma rígida (22:10). Vamos a repetir el peso muy rápidamente. Y luego pasaremos al siguiente tema (22:17).

Así que voy a ir aquí. Voy a usar def height; es decir, con prct, (22:25) obtendremos de nuevo el porcentaje. Y diremos return, abriendo paréntesis para que podamos (22:32) tener una visión más clara de esta fórmula.

Y podemos decir que `settings dot` es dividido entre 100 (22:42) multiplicado por el porcentaje. Así que ahora puedo continuar y hacer prácticamente lo mismo con los diferentes (22:50) valores predefinidos aquí. Veamos.

En el fotograma superior, usamos toda la ventana. (22:56) Así que tiene sentido cambiar esto solo al valor de `settings dot width`, correcto, porque (23:01) ya es 1440. Pasemos al fotograma izquierdo.

En el fotograma izquierdo, vemos que usamos el 25% del (23:10) ancho total. Podemos usar `utils dot height PRC t`. Disculpen, con `utils dot height PRC t`, cambiaré eso (23:19). Y luego pasaré 25.

También pueden ver que en la altura, usamos el 75% de nuestra altura total (23:30). Esto significa que puedo usar `utils dot height PRC t` y el año de paso (23:38) 75. Y puedo hacer lo mismo cuando quiero ubicar mi fotograma izquierdo.

Observen cómo el valor del eje y (23:46) aquí es nuevamente el 25% de la altura total. Así que puedo decir `utils dot height PRC t` y pasar aquí (23:56) 25%. Ahora veamos si hemos hecho un buen trabajo.

Como pueden ver, el resultado es nuevamente perfecto. (24:03) Bien, ahora que tenemos esto, permítanme cambiarlos a los valores originales y continuar (24:09) desde aquí para crear nuestro último fotograma. Dijimos que también nos gustaría tener un fotograma más (24:15) que estará en el centro de nuestra página, para que tengamos un fotograma dedicado (24:21) a nuestro juego, ¿cierto?

Así que rápidamente crearé un fotograma central que será igual (24:27) a un fotograma. Este debería estar dentro de la ruta, el fondo debería ser quizás verde, para que podamos (24:35) separarlo. Tendremos que cambiar todo a negro en unos minutos. Y luego (24:43) quiero decir que el ancho es igual al ancho del punto de utilidad PRC t. Y esta vez, me gustaría usar el 75% de nuestro ancho total. (24:51) La altura será, de nuevo, utils dot height PRC t, y también será del 75 %. Ahora (25:00) el método place recibirá los siguientes argumentos. El valor de x debería ser utils dot (25:10) width PRC t 25 %.

Porque queremos empezar desde aquí. Y la altura, es decir, el eje y (25:19), debería ser la misma. Voy a duplicarlo.

Y cambiar esto a (25:26) height PRC t, así, porque queremos que este empiece desde aquí. Tomará (25:33) 360 píxeles desde la izquierda y 180 píxeles desde la parte superior. Si probáramos nuestro código (25:43), verían que tenemos el resultado esperado. Ahora es el momento perfecto para (25:49) cambiar nuestros marcos a negro, porque hemos hecho un gran trabajo. Podemos cambiarlos a (25:54) negro. Lo que verán es una ventana que parece completamente negra. (26:01) Pero en el fondo, en el código Python real, tenemos dos marcos divididos tal como (26:08) queríamos.

Ahora que he cambiado todos los fondos a negro, veremos (26:13) lo mismo. Pero en el fondo, en el código original, tenemos dos marcos, (26:19) o sea, tres marcos. Bien, espero que hayan disfrutado desarrollando este proyecto hasta ese punto (26:24).

En el próximo episodio, aprenderemos a crear más elementos en esa ventana. Así (26:31) podremos hacer clic en elementos para tener un juego básico en marcha. Espero verte (26:37) en el próximo episodio. Si te gustó, no olvides darle a "Me gusta" (26:41) y suscribirte para no perderte ningún video. Nos vemos muy pronto.