

Cerrado

Abrió Hace 2 añospor  **Jasper Koehorst**

Aceptar código de salida específico

Al ejecutar el gitlab runner y realizar algunas pruebas, se supone que se debe devolver un código de salida 64 y el gitlab runner lo ve como un error y, por lo tanto, deja de compilar. Esto tiene sentido, pero cuando trato de establecer el código de salida en 0 después de ejecutar el paquete jar, indica que el código de salida es 1.

Por ejemplo:

```
#test the gradle build
testbuild:
  stage: test
  script:
    # Temp skipping test cases, remove afterwards
    - gradle build --info -x test
    # The first echo $? should be 64 and the second echo should give 0 (works in local terminal)
    - java -jar ./build/libs/signalp.jar; echo $?; echo $?
    # This command should check if the exit code is 64 and then should set to 0
    - java -jar ./build/libs/signalp.jar; if [ $? -eq 64 ]; then echo "success"; (exit 0); fi
```



Sin embargo, ambos métodos dan como resultado un

```
ERROR: Job failed: exit code 1
```

¿Alguna idea de cómo aceptar un código de salida específico para una línea o cómo establecer el resultado en 0 después de ejecutar el jar?

Problemas vinculados 

 0 0






 **Alessio Caiazza**  [@nolith](#) · Hace 2 años Mantenedor


Hola [@jjkoehorst](#)

¿Puedes probar esto, por favor?

```
java -jar ./build/libs/signalp.jar || if [ $? -eq 64 ]; then echo "success"; fi
```

Tenga en cuenta que si `java -jar ./build/libs/signalp.jar` devuelve `0` su tubería será verde también.

-  **Alessio Caiazza**  [@nolith](#) agregó etiqueta de `solicitud de soporte` [Hace 2 años](#)
-   **GitLab Bot**  [@gitlab-bot](#) agregó la etiqueta `CI / CD [DEPRECATED]` [Hace 2 años](#)

 **Jasper Koehorst** [@jjkoehorst](#) · Hace 2 años

Hola [@nolith](#)

He intentado lo siguiente y al principio parecía funcionar. Pero ahora creo que captura todos los códigos de salida como ok:

```
BUILD SUCCESSFUL in 3m 36s
4 actionable tasks: 4 executed
Stopped 0 worker daemon(s).
$ java -jar ./build/libs/signalp.jar || if [ $? -eq 64 ]; then echo "success"; fi
Error: Could not find or load main class nl.wur.ssb.signalp.Appppppp
Job succeeded
```

Si ejecuto el jar en mi computadora, devuelve un código de salida 1

```
→ signalp git:(dev) X java -jar signalp.jar
Error: Could not find or load main class nl.wur.ssb.signalp.Appppppp
```

```
Caused by: java.lang.ClassNotFoundException: nl.wur.ssb.signalp.Appppppp
→ signalp git:(dev) X echo $?
1
```

editar:

Cuando ejecuto el comando completo en mi computadora, también devuelve 0:

```
java -jar ./build/libs/signalp.jar || if [ $? -eq 64 ]; then echo "success"; fi
echo $?
0
```

También he intentado realizarlo de esta manera:


```
java -jar ./build/libs/signalp.jar; if [ $? -ne 64 ]; then echo "failure"; (exit 11) fi
ERROR: Job failed: exit code 1
```

Mientras que localmente devuelve 11

Acerca de devolver el código de salida 0. Algunas canalizaciones prefieren que cuando se llame incorrectamente al programa o sin argumentos, se muestre una ayuda con un código de salida distinto de cero. Por lo tanto, se eligió 64 como significa, `command line usage error` pero para probar sería bueno ver si obtenemos la descripción general de la ayuda adecuada y, por lo tanto, debemos aceptar el código de salida de 64 o establecerlo luego en 0.

Editado por [Jasper Koehorst](#) Hace 2 años



Alessio Caiazza  [@nolith](#) · [Hace 2 años](#)

Mantenedor

[@jjkoehorst](#) Personalmente, creo que su script es demasiado complejo para quedarse `.gitlab-ci.yml` y puede merecer un archivo dedicado en su repositorio, pero es una cuestión de gusto personal.

Todos los trabajos de CI se ejecutan, lo `set -eo pipefail` que significa que falla si alguno de sus pasos falla, incluso aquellos en una tubería.

La única forma de evitar tal falla es usarla, `|| something-else` pero de esta manera está perdiendo toda la falla legítima.

Ahora, si desea permanecer con todo lo `.gitlab-ci.yml` que necesita, debe agregar una `else` causa `if` que garantice la falla, es decir (no probado):

```
java -jar ./build/libs/signalp.jar || if [ $? -eq 64 ]; then echo "success"; else exit
```



Jasper Koehorst [@jjkoehorst](#) cerrado [Hace 2 años](#)



Daniel Herrmann [@waza-ari](#) · [hace 1 año](#)

Gracias por este post Me gustaría ir con una solución ligeramente diferente, ya que `exit 1` mientras no devuelvo el código de salida 1 sino que simplemente salgo de la ejecución del contenedor. Podría ser una pequeña diferencia, pero prefiero ir con:

```
ar ./build/libs/signalp.jar || if [ $? -eq 64 ]; then echo "success"; else /bin/false; fi
```

`/bin/false` devuelve el código de salida 0. Ejecutando en un contenedor alpino de vainilla:

```
# Normal run
/ # ls /asd
ls: /asd: No such file or directory
/ # echo $?
1

# Catch exit code 1
/ # ls /asd || if [ $? -eq 1 ]; then echo "success"; else /bin/false; fi
ls: /asd: No such file or directory
success
/ # echo $?
```

```
0
```

```
# Catching exit code 2 yields the following
/ # ls /asd || if [ $? -eq 2 ]; then echo "success"; else /bin/false; fi
ls: /asd: No such file or directory
/ # echo $?
1
```

Incluso podría ser mejor pasar el código de salida original, pero no descubrí cómo hacerlo.



Maksym Lushpenko @maksym.lushpenko · hace 3 meses

@waza-ari gracias por tus fragmentos, esto parece funcionar:

```
/ # (exit 20) || code=$?; if [ $code -eq 19 ]; then echo "success"; else echo $code; fi
20
/ # (exit 20) || code=$?; if [ $code -eq 20 ]; then echo "success"; else echo $code; fi
success
```

Por favor [regístrese](#) o [inicie sesión](#) para responder