Logs de estabilidad

Last updated by | Walter Arboleda Castañeda | Nov 12, 2024 at 3:02 PM GMT-5

Objetivo

Crear archivos logs que evidencien la estabilidad de una rutina al ejecutarla con el 100% de los datos y que no se presenten incidentes en cuanto a consumo de recursos en la plataforma o fallas en el proceso general.

Configuración

A continuación se describe el paso a paso para la generación de un log de estabilidad.

1. Realice las configuraciones necesarias en el archivo de parámetros tal cual como la rutina debe ejecutar en producción.
2. Configure un ambiente python de acuerdo a la versión instalada en los ambientes donde será desplegada la rutina.
3. Configure los parámetros para la generación y alamacenamiento de los logs de estabilidad en la carpeta logs_compilación

- Los logs de compilación deben almacenarse en la carpeta logs_compilación en la raiz del repositorio. Para esto puede utilizar las siguientes lineas de código que identifican el directorio actual y crean la carpeta

```
import pkg_resources
# Identificar el directorio actual
path = pkg_resources.resource_filename(__name__, "")

# Ruta de almacenamiento de logs
logs_path = os.path.join(path.split("src")[0], "logs_calendarizacion")

# Crear la carpeta logs_calendarización
if not os.path.exists(logs_path):
    print("No existía la carpeta de logs para calendarización, se está creando.")
    os.mkdir(logs_path)
```

- El orquestador2 identificará esta configuración mediante los parámetros globales, por tanto se pueden ingresar mediante el archivo `config.json`, o ingresarlos como parámetros adicionales en el archivo de `ejecucion.py`. **En ambas configuraciones se debe recordar configuarar de nuevo el paquete al estado en el que será ejecutado en produccion luego de finalizar este proceso**.
- Ingrese la ruta de almacenamiento de los logs mediante el parámetro `log_path`.

```
# k: Diccionario de parámetros globales
kw["log_path"] = logs_path
```

- Ingrese el tipo de log que va a generar, en este caso estabilidad (est)

```
    # k: Diccionario de parámetros globales
    kw["log_type"] = "est"
```

Nota: Puede utilizar los controles en tiempo real para la ejecución, pero no es permitido el uso del parametro ***`porcentaje_limit`***

3. Para la ejecución de la rutina puede:

- Realizar una instalación local en su maquina, esto puede relizarse ubicando la ruta donde se encuentra su archivo setup.py o setup.cfg (sin incluir el nombre del archivo) y ejecute

  `pip install <path_setup>` donde `<path_setup>` es la ruta en su maquina

- Ejecute la rutina con el comando

  `python -m nombre_paquete.ejecucion` Recuerde añadir los parámetros adicionales por consola si así lo ha configurado

- Puede realizar una ejecución del proceso mediante un proceso debug o simulando los parámetros mediante una ejecución tradicional del archivo `ejecucion.py` .

Salida

Cada ejecucion generará un archivo est_.log. Genere los logs que sean necesarios según las políticas y contratos de calendarización.

```
paquete-xyz
|
|- .gitignore
|- .pypirc
|- MANIFEST.in
|- README.md
|- script_ejecucion.py
|- setup.cfg
|- setup.py
|
└── logs_calendarizacion
|    └── - est_...log
|        |- est_...log
|        |- est_...log
|
└── src
    |
    └── paquete_xyz
        |
        |- ...
```

Ejemplos de configuraciones para generar log de estabilidad

- Configuraición en archivo ejecución.py

```
##################################
# Generación Logs - Estabilidad
##################################
path = pkg_resources.resource_filename(__name__, "")
logs_path = os.path.join(path.split("src")[0], "logs_calendarizacion")
kw["log_path"] = logs_path
kw["log_type"] = "est"

if not os.path.exists(logs_path):
    print("No existía la carpeta de logs para calendarización, se está creando.")
    os.mkdir(logs_path)
##################################

# Paso de los parámetros a los step y al orquestador
steps = [ Step_1(**kw) , Step_2(**kw) , ...]
orquestador = Orchestrator('Orch2-Ejemplo', steps , **kw)
```