EBook Gratis

APRENDIZAJE robotframework

Free unaffiliated eBook created from
Stack Overflow contributors.

#robotframework

work

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: robotframework

It is an unofficial and free robotframework ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official robotframework.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con robotframework

Observaciones

Esta sección proporciona una descripción general de qué es el marco robot, y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro del marco de robot, y vincular a los temas relacionados. Dado que la Documentación para robotframework es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Versiones

Versión	Fecha de lanzamiento
Robot Framework 3.0.2	2017-02-14
Robot Framework 3.0.1	2017-01-06
Robot Framework 3.0	2015-12-31
Robot Framework 2.9.2	2015-10-09
Robot Framework 2.9.1	2015-08-28
Robot Framework 2.9	2015-07-30

Examples

Instalación o configuración

Instrucciones detalladas sobre cómo configurar o instalar Robot Framework.

Robot framework es un marco de automatización de prueba genérico. Esto se implementa utilizando Python y es compatible con Python 2 y Python 3 Jython (JVM) y IronPython (.NET) y PyPy. por

1. Test de aceptación
2. Desarrollo guiado por pruebas de aceptación (ATDD)

Prerrequisitos

1. Instalar un intérpretes
2. Configurando la ruta

3. Configuración de https_proxy

Python tiene las implementaciones más avanzadas y se sugiere usar Python, si no tiene requisitos excepcionales.

Versión de Robot Framework	Versión de intérprete soportada
Robot Framework 3.0	Python 2.6
Robot Framework 3.0	Python 2.7
Robot Framework 3.0	Python 3.3
Robot Framework 3.0	Jython 2.7 y Java 7
Robot Framework 3.0	IronPython 2.7
Robot Framework 2.5-2.8	Python 2.5
Robot Framework 2.5-2.8	Jython 2.5
Robot Framework 2.0-2.1	Python 2.3
Robot Framework 2.0-2.1	Python 2.4
Robot Framework 2.0-2.1	Jython 2.2

Instalación de Python

La versión deseada de python se puede descargar desde https://www.python.org/downloads/

Instalacion jython

Puede encontrar un instalador en http://jython.org . Puede ejecutar este paquete JAR ejecutable desde la línea de comandos como javaava -jar jython_installer-.jar.

Instalación de IronPython

Se puede encontrar un instalador en http://ironpython.net/download/ para IronPython 2.7.Cuando se usa IronPython, una dependencia adicional es la instalación del módulo elementtree 1.2.7

Configurando PATH y configurando https_proxy

Agregue el directorio de instalación de Python (de manera predeterminada C: \ Python27, C: \ Python27 \ Scripts, C: \ jython2.7.0 \ bin etc. en Windows) y el directorio de Scripts al principio de su variable de ruta

El valor de https_proxy debe ser la URL del proxy. Esto es necesario cuando estos paquetes se instalan con pip y usted está en una red proxy

Instalación de Robot Framework con pip

Instala la última versión de robotframework

```
pip install robotframework
```

Instalar una versión específica

```
pip install robotframework==2.0
```

Instalación de Robot Framework desde la fuente

La distribución de origen de Robot Framework se puede encontrar en https://code.google.com/archive/p/robotframework/downloads.Robot Framework se instala desde la fuente utilizando el script setup.py estándar de Python en el directorio de scripts de origen

```
python setup.py install
jython setup.py install
ipy setup.py install
```

Instalar Robot Framework 3.0 en una máquina con Windows usando Python 2.7.11

Esta es una guía rápida para que Robot Framework 3.0 funcione en una máquina con Windows usando Python 2.7.11. No profundiza demasiado en el por qué y en cómo, simplemente lo pone en funcionamiento. ¡Lo primero es lo primero, no nos vamos a instalar Python!

1. Descargar Python 2.7.11 para Windows . (Instalador de MSI para Windows x86-64 o instalador de MSI para Windows x86 según la arquitectura)

2. Ejecute la instalación, asegúrese de instalar "pip" y de optar por "Agregar python.exe a la ruta" (es posible que deba reiniciar su máquina para aprovechar la ruta de Python. En esta guía, se supone que no tengo ese lujo)

3. Una vez que esté instalado, hagamos una revisión rápida para asegurarnos de que se instaló correctamente. Ejecute CMD como administrador y navegue hasta donde Python se instaló en `cd C:\Python27` y escriba `python -V` . Debe devolver "Python 2.7.11"

Eso es todo, Python ahora está instalado en tu máquina. La siguiente parte es obtener Robot Framework Installed en su máquina usando pip.

1. Primero, asegurémonos de tener la última versión de pip, primero navegando al directorio de scripts en Python `cd C:\Python27\Scripts` y luego ingresando `python -m pip install -U pip` . ¡Debería decir que tienes la versión más actualizada instalada!

2. A continuación, `pip install robotframework` Robot Framework ingresando `pip install robotframework`

3. Una vez que pip haya terminado de descargar e instalar los archivos, ingrese `robot --version` para asegurarse de que se instaló correctamente. Debería decir Robot Framework 3.0 (Python 2.7.11 en win32 / 64)

4. (Opcional) Si en el futuro hay una actualización para Robot Framework, puede ejecutar este comando `pip install --upgrade robotframework`

Lea Empezando con robotframework en línea:
https://riptutorial.com/es/robotframework/topic/5187/empezando-con-robotframework

Capítulo 2: ¿Cómo se utiliza el marco de robot en las pruebas de automatización en sistemas integrados?

Introducción

Robot framework es ampliamente utilizado en las pruebas de automatización de productos integrados. Tomaremos un producto incorporado como ejemplo y veremos cómo automatizar los casos de prueba utilizando Robot Framework.

Observaciones

Abreviatura:

- RPS - Fuente de alimentación remota
- RF - Robot marco de trabajo

Examples

Prueba de alimentación remota

Simulación de alimentación remota

Como no tenemos un hardware de fuente de alimentación remota real, lo simularemos usando el programa python.

Idea básica sobre RPS

- En realidad, la fuente de alimentación remota tiene un servidor http.
- El usuario puede enviar comandos para encender / apagar la fuente de alimentación usando la solicitud http.

Vamos a simular la fuente de alimentación remota utilizando el siguiente programa rps-server.py.

```
from flask import Flask, request
from flask_httpauth import HTTPBasicAuth


app = Flask(__name__)
auth = HTTPBasicAuth()

users = {
        'admin': '12345678'
}
```

```
app.url_map.strict_slashes = False

PINS = ['P60', 'P61', 'P62', 'P63']

PINS_STATUS = {'P60':'0', 'P61': '0', 'P62':'0', 'P63':'0'}

@auth.get_password
def get_pw(username):
        if username in users:
                return users.get(username)
        return None

@app.route('/')
@auth.login_required
def index():
        return "Hello, %s!" % auth.username()

def get_html_string():
        html_str = '<html>P60={}P61={}P62={}P63={}</html>'.format(PINS_STATUS['P60'],
                                                PINS_STATUS['P61'],
                                            PINS_STATUS['P62'],
                                            PINS_STATUS['P63'])
        return html_str

def parse_cmd_args(args):
        global current_status
        if str(args['CMD']) == 'SetPower':
                for key in args:
                        if key in PINS:
                                PINS_STATUS[key] = str(args[key])

                return get_html_string()

        if str(args['CMD']) == 'GetPower':
                return get_html_string()


    @app.route('/SetCmd', methods=['GET','POST'])
    def rps():
        if request.method=="GET":
                args=request.args.to_dict()
                ret = parse_cmd_args(args)
                return ret
```

El código anterior realmente simula el servidor http para controlar la fuente de alimentación remota.

¿Cómo ejecutar el servidor RPS?

```
$ export FLASK_APP=rps-server.py
$ flask run
```

¿Cómo enviar comandos al servidor rps?

Los siguientes son los dos comandos utilizados para controlar el RPS

1. SetPower
2. Obtener energía

Por defecto el servidor estará escuchando en el puerto 5000.

Los puertos de alimentación son,

1. P60
2. P61
3. P62
4. P64

Los estados de los puertos son,

1. EN 1
2. OFF - 0

Requerimientos

Los requisitos para construir una fuente de alimentación remota son

1. La fuente de alimentación remota debe poder encenderse / apagarse remotamente
2. Se puede acceder de forma remota al estado de la fuente de alimentación remota.

Casos de prueba derivados

Casos de prueba derivados del requisito

1. Encienda la fuente de alimentación 2 de forma remota.
2. Verifique que la fuente de alimentación 2 esté encendida.
3. Apague la fuente de alimentación 2 de forma remota.
4. Verifique que la fuente de alimentación 2 esté apagada.

Prueba manual

- Ejecutar el servidor rps.
- Para activar el puerto 3, abra un navegador y proporcione la siguiente URI

```
http://admin:12345678@localhost:5000/SetCmd?CMD=SetPower&P62=1
```

- Para obtener el estado de todos los puertos.

```
http://admin:12345678@localhost:5000/SetCmd?CMD=GetPower
```

Biblioteca de prueba de escritura

Necesitamos escribir una biblioteca de prueba en python para enviar comandos http utilizando la

solicitud http. Más adelante usaremos esta biblioteca como palabras clave en el trabajo de marco de robot.

comandos.py

Vamos a utilizar la biblioteca de commands.py para enviar SetPower y GetPower.

```
import requests
import re


class commands(object):

    ROBOT_LIBRARY_SCOPE = 'GLOBAL'
    def __init__(self, ip='localhost:5000'):
        self.ip_address = ip
        self.query = {}
        self.user = 'admin'
        self.passw = '12345678'

    def form_query(self, state, cmd, port):
        port = self.get_port_no(port)
        self.query = {port: state}
        return self.query

    def get_port_no(self, port_no):
        port = 'P6' + str(port_no)
        return port

    def clean_html(self, data):
        exp = re.compile('<.*?>')
        text = re.sub(exp, "", data)
        return text.rstrip()

    def send_cmds(self, cmd, port=None, state=None):
        url = 'http://{}:{}@{}/SetCmd?CMD={}'\
              .format(self.user,
                      self.passw,
                      self.ip_address,
                      cmd)
        print url
        if cmd == 'SetPower':
            self.form_query(state, cmd, port)
            self.req = requests.get(url, params=self.query)
            return True
        elif cmd == 'GetPower':
            self.req = requests.get(url)
            data = self.clean_html(self.req.text)
            return data
        else:
            return False

        return self.req.text


# c = commands('localhost:5000')

# c.send_cmds('SetPower', 2, 1)
# c.send_cmds('SetPower', 3, 1)
```

```
# print c.send_cmds('GetPower')
```

Documentación de la palabra clave de Python

1. `send_cmds(cmd, port=None, state=None)` es la función que vamos a utilizar.
2. Al utilizar esta función en la palabra clave de Robot, no hay necesidad de preocuparse por _ , `Lowercaser` o `Uppercase` en el nombre de la función.

La función Python se verá así mientras se usa como palabra clave,

```
Send Cmds       cmd   port  state
```

Prueba de escritura Palabras clave

Vamos a utilizar `Send Cmds` como palabra clave de python en el conjunto de pruebas.

- Los comandos de envío RPS utilizan los siguientes cuatro argumentos para establecer el poder
    - comando = SetPower
    - puerto = 2
    - estado = 1 para ON / 0 para off Cuando llamamos a ese comando, se encenderá / apagará la fuente de alimentación
- RPS get power devolverá el estado de todos los puertos de fuente de alimentación

```
*** Keywords ***
RPS send commands
    [Arguments]    ${command}    ${port}    ${state}
    ${output}=    Send cmds    ${command}  ${port}  ${state}
    [return]    ${output}

RPS get Power
    [Arguments]    ${command}
    ${output}=    Send cmds    ${command}
    [return]    ${output}}
```

Algoritmo para probar la fuente de alimentación

1. Establecer la potencia a un puerto
2. Compruebe el estado de cmd
3. Obtenga el estado del puerto y verifique si está ENCENDIDO / APAGADO

Escribir casos de prueba utilizando las palabras clave anteriores.

Ahora estamos listos para escribir un caso de prueba usando las siguientes dos palabras clave

- Comandos de envío RPS: para configurar y deshabilitar una potencia de puerto
- RPS obtiene energía - Para obtener el estado de todo el puerto

```
*** Settings ***
Library      commands.py

*** Test Cases ***
Turn on Power supply 2 remotely
     ${out}=    RPS send commands     SetPower  2  1
     Should be equal    ${out}  ${True}

Verify power supply 2 is on
    ${out}=     RPS get power    GetPower
    should contain    ${out}  P62=1

Turn off Power supply 2 remotely
     ${out}=    RPS send commands     SetPower  2  0
     Should be equal    ${out}  ${True}

Verify power supply 2 is off
    ${out}=     RPS get power    GetPower
    should contain    ${out}  P62=0
```

Crear un nombre de archivo `remote-power-supply.robot`

Copie las palabras clave anteriores y el caso de prueba en el archivo.

¿Cómo ejecutar el servidor RPS y remote-power-supply.robot?

- Ejecutar primero la fuente de alimentación remota
- Ejecute el conjunto de prueba remote-power-supply.robot

```
$ export FLASK_APP=rps-server.py
$ flask run
$ pybot remote-power-supply.robot
```

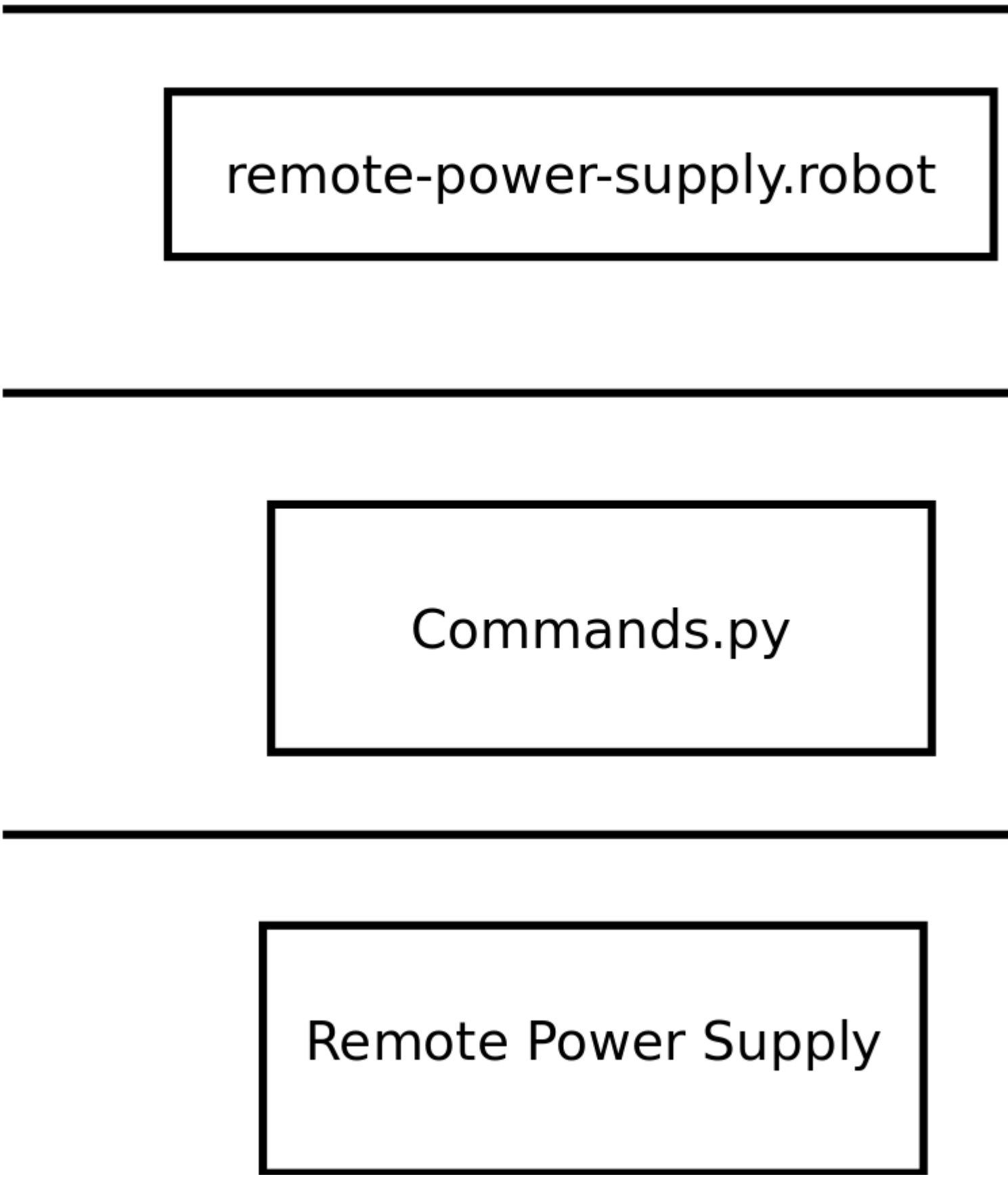
Salida

```
$ pybot remote-pwer-supply.robot
==============================================================================
Remote-Pwer-Supply
==============================================================================
Turn on Power supply 2 remotely                                       | PASS |
------------------------------------------------------------------------------
Verify power supply 2 is on                                           | PASS |
------------------------------------------------------------------------------
Turn off Power supply 2 remotely                                      | PASS |
------------------------------------------------------------------------------
Verify power supply 2 is off                                          | PASS |
------------------------------------------------------------------------------
Remote-Pwer-Supply                                                    | PASS |
4 critical tests, 4 passed, 0 failed
```

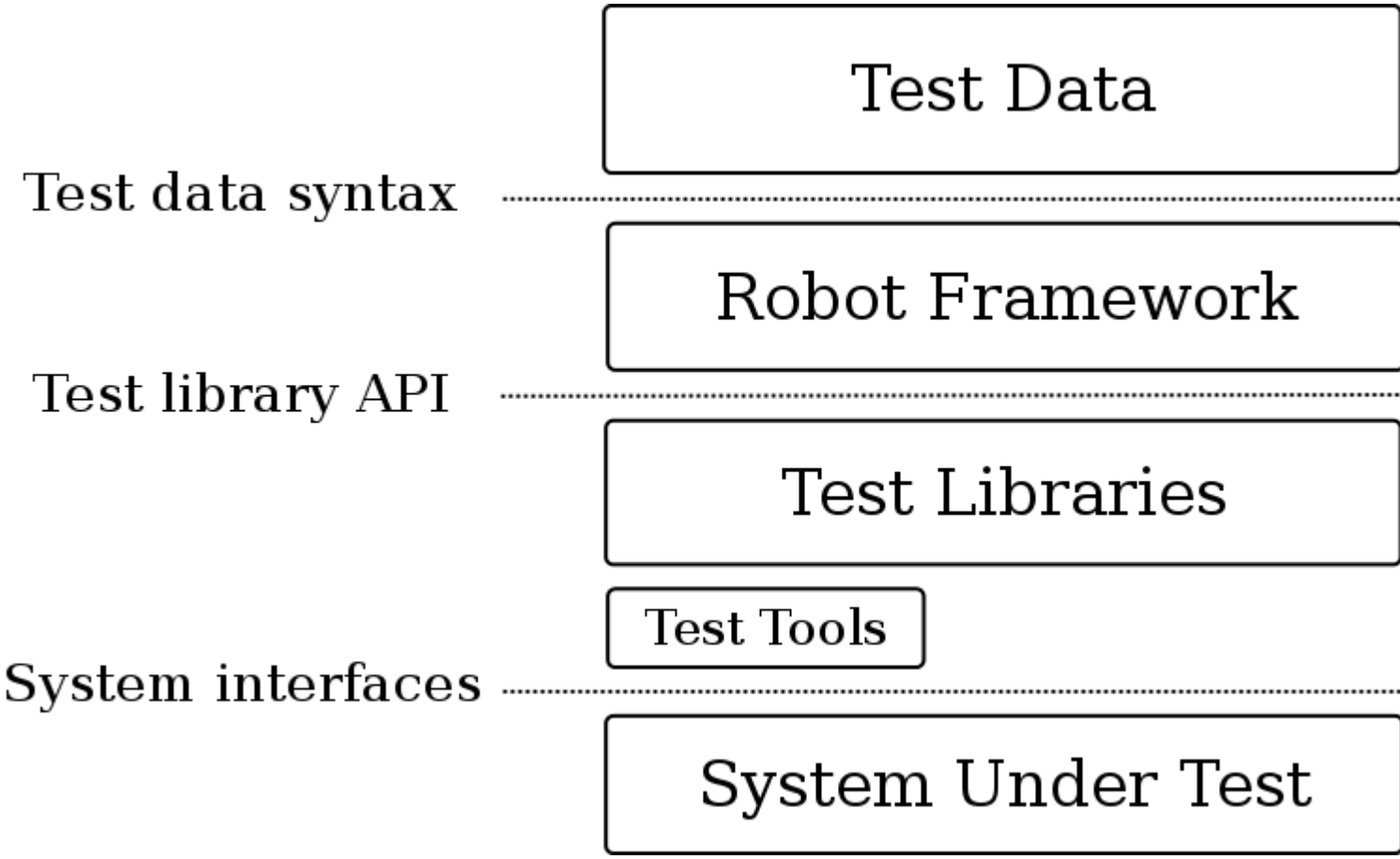
```
4 tests total, 4 passed, 0 failed
==============================================================================
Output:  /tmp/talks/robot-framework-intro/test-cases/output.xml
Log:     /tmp/talks/robot-framework-intro/test-cases/log.html
Report:  /tmp/talks/robot-framework-intro/test-cases/report.html
```

Los siguientes dos diagramas explican la arquitectura de prueba entre RPS y RF

Arquitectura de prueba de alimentación remota


commands, state, port no
remote-power-supply.robot
Commands.py
Remote Power Supply

Robot de marco de trabajo de arquitectura.



Creditos

Gracias al robot framework para el diagrama de arquitectura.

El código completo está disponible aquí.

Fuente de alimentación remota

comandos.py

fuente de alimentación remota.

Lea ¿Cómo se utiliza el marco de robot en las pruebas de automatización en sistemas integrados? en línea: https://riptutorial.com/es/robotframework/topic/10672/-como-se-utiliza-el-marco-de-robot-en-las-pruebas-de-automatizacion-en-sistemas-integrados-

Creditos

S. No	Capítulos	Contributors
1	Empezando con robotframework	Community, Goralight, Hariprasad, Shijo
2	¿Cómo se utiliza el marco de robot en las pruebas de automatización en sistemas integrados?	kame, sakthirengaraj