Starting and stopping

- Main commands to launch and check
- All the ways to start WCS
  - Launch as service
    - Automatic service restart on failure
  - Launch from command line
- Environment variables setup
- Starting with stdout output
  - Starting with JVM output redirection to a log file
- Launching with different user permissions
  - Launching builds 5.2.864-5.2.972
  - Launching build 5.2.976 and newer
  - Switching launch mode
  - Folder permissions setting when starting from flashphoner user
- JVM parameters
  - Java version automatic detection and JVM parameters correction
- Automatic WCS health checking after launch

Main commands to launch and check

After you activate the license, start WCS using the command:

```
sudo systemctl start webcallserver
```

Stopping the server is done with the command

```
sudo systemctl stop webcallserver
```

Here are a few ways to make sure the server has started and is ready to work:

1.Make sure the server process is running.

```
pgrep -afn com.flashphoner.server.Server
```

The console should display WCS Core process (PID 6880 on the example below):

```
[root@localhost ~]# pgrep -afn com.flashphoner.server.Server
6880 java -Xmx4g -Xms4g -XX:+UseConcMarkSweepGC -XX:+UseCMSInitiatingOccupancyOnly -XX:
CMSInitiatingOccupancyFraction=70 -Djava.net.preferIPv4Stack=true -Dcom.sun.management.jmxremote=true -Dcom.sun.
management.jmxremote.local.only=false -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.
authenticate=false -Dcom.sun.management.jmxremote.port=50999 -Djava.rmi.server.hostname=p13.flashphoner.com -XX:
ErrorFile=/usr/local/FlashphonerWebCallServer/logs/error%p.log -XX:+PrintGCDateStamps -XX:+PrintGCDetails -
Xloggc:/usr/local/FlashphonerWebCallServer/logs/gc-core-2021-06-25_14-44.log -XX:+ExplicitGCInvokesConcurrent -
Dsun.rmi.dgc.client.gcInterval=36000000000 -Dsun.rmi.dgc.server.gcInterval=36000000000 -Dcom.flashphoner.fms.
AppHome=/usr/local/FlashphonerWebCallServer -Djava.library.path=/usr/local/FlashphonerWebCallServer/lib/so:/usr
/local/FlashphonerWebCallServer/lib -DWCS_NON_ROOT=true -DsessionDebugEnabled=false -Djdk.tls.client.protocols="
TLSv1,TLSv1.1,TLSv1.2" -cp /usr/local/FlashphonerWebCallServer/lib/* com.flashphoner.server.Server
[root@localhost ~]#
```

2. Make sure the server process listens the main ports.

```
netstat -nlp | grep java

[root@localhost tmp]# netstat -nlp | grep java
tcp        0      0 0.0.0.0:1098            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:1935            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 127.0.0.1:2001          0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8081            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 127.0.0.1:2002          0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8082            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 192.168.1.5:3478       0.0.0.0:*              LISTEN      6880/java
tcp        0      0 0.0.0.0:50999           0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8888            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8443            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8444            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:8445            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:9091            0.0.0.0:*               LISTEN      6880/java
tcp        0      0 0.0.0.0:45731           0.0.0.0:*               LISTEN      6880/java
udp        0      0 0.0.0.0:1935            0.0.0.0:*                           6880/java
```

If you used default ports settings, you should see ports 8080, 8444 (Websockets) and 1935 (RTMP) as well as other ports you configured for the WCS server in the list.

3. Make sure the WCS server writes the main server log

```
tail -f /usr/local/FlashphonerWebCallServer/logs/server_logs/flashphoner.log
```

The log should display information about settings the server started with.

For example:

```
18:29:51,945 INFO        SettingsLoader - main Override setting allow_outside_codecs: from true to false
18:29:51,974 INFO        SettingsLoader - main Override setting codecs: from null to opus,alaw,ulaw,g729,
speex16,g722,mpeg4-generic,telephone-event,h264,vp8,flv,mpv
18:29:51,975 INFO        SettingsLoader - main Override setting media_port_from: from 31001 to 31001
18:29:51,978 INFO        SettingsLoader - main Override setting keep_alive.enabled: from websocket,rtmp,rtmfp
to websocket,rtmfp
18:29:51,978 INFO        SettingsLoader - main Override setting webrtc_cc_min_bitrate: from 30000 to 3000000
18:29:51,979 INFO        SettingsLoader - main Override setting codecs_exclude_sip: from null to mpeg4-generic,
flv,mpv,opus,ulaw,h264,g722,g729
18:29:51,979 INFO        SettingsLoader - main Override setting wss.port: from 8443 to 8443
18:29:51,980 INFO        SettingsLoader - main Override setting codecs_exclude_sip_rtmp: from null to opus,g729,
g722,mpeg4-generic,vp8,mpv
18:29:51,980 INFO        SettingsLoader - main Override setting codecs_exclude_streaming: from null to
telephone-event
18:29:51,980 INFO        SettingsLoader - main Override setting webrtc_cc_max_bitrate: from 10000000 to 7000000
18:29:51,980 INFO        SettingsLoader - main Override setting ip: from 0.0.0.0 to 192.168.1.5
18:29:51,980 INFO        SettingsLoader - main Override setting client_log_level: from INFO to DEBUG
18:29:51,980 INFO        SettingsLoader - main Override setting ip_local: from 0.0.0.0 to 192.168.1.5
18:29:51,980 INFO        SettingsLoader - main Override setting media_port_to: from 32000 to 32000
18:29:51,981 INFO        SettingsLoader - main Override setting ws.port: from 8080 to 8080
```

Logs should react on connection of web clients. If that does not happen during testing, make sure the server process is running and the web client is configured properly to connect this particular server. See the Troubleshooting section for additional information.

If the server process is running and logs have no error, this means the WCS server is ready to work and you can start testing it.

All the ways to start WCS

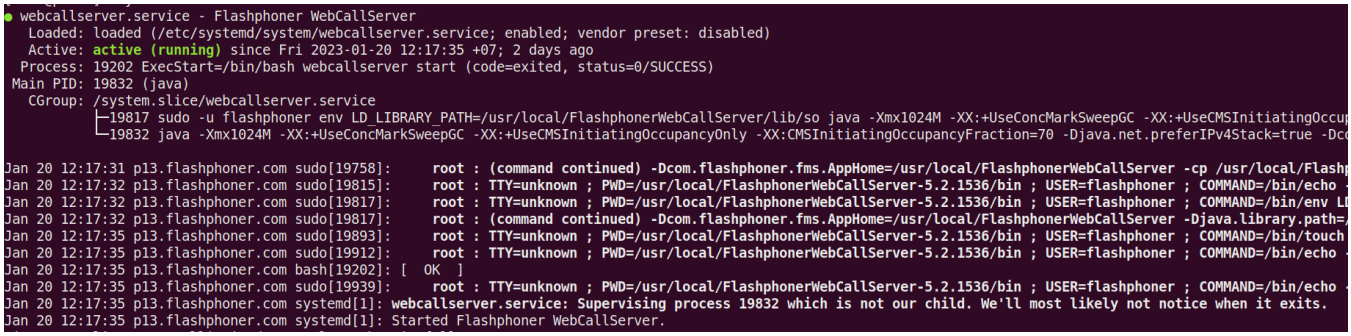
Launch as service

Use the following command to launch WCS as service:

```
sudo systemctl start webcallserver
```

This is the preferrable way to start WCS. In this case, the service will start from `root` user, and the main WCS process will start from `flashphoner` or `root` user depending on launch mode configuration.

Since build 5.2.1537 the service type is changed from `simple` to `forking`. Also the file permissions containing the main WCS process PID are set according to systemd requirements. The service is marked as `active (running)`, and systemd may send a signaldirectly to the main WCS process to stop it if necessary



Automatic service restart on failure

Since build 5.2.1562 the service unit `webcallserver.service` will be restarted automatically if the service becomes `failed` for some reason. Up to 5 relaunch tries will be done if no more than 2 minutes passed between them.

The service still may be stopped or started manually. If service is stopped manually, it will not be relaunched.

Automatic service restart n=may be disabled with the following command

```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./webcallserver set-service-restart disable
```

Launch from command line

WCS can be started from command line if necessary:

```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./webcallserver start
```

This way to start is useful for testing and debugging purposes.

Environment variables setup

Environment variables and parameters of the start are set in the `setenv.sh` script. In this script you can use additional parameters for WCS Core and WCS Manager. Also, here you can set the parameter that prevent memory leaks on multi-CPU systems:

```
MALLOC_ARENA_MAX=4
```

Starting with stdout output

In some cases, for example, if the server won't start and does not produce any errors, you may need to start the server with direct logging to the `stdout` console. Direct output to `stdout` cannot be used in production, because the server will be stopped if the console is closed or the SSH connection is lost. That is why we recommend using `stdout` output only for debug purposes.

To start the server in this mode, use the following command:

```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./webcallserver start standalone
```

Starting with JVM output redirection to a log file

Since build 5.2.1562 WCS may be launched with JVM output redirected to a file like direct `stdout` logging

```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./webcallserver start --java-log
```

In this case, all Java output will be redirected to the `/usr/local/FlashphonerWebCallServer/logs/java.log` file. The feature should be used for debugging purposes only but not in production, because the resulting file size may be too large.

Launching with different user permissions

Launching builds 5.2.864-5.2.972

Since build5.2.864, the permissions to launch WSC are defined as follows:

1. The command

```
sudo systemctl start webcallserver
```

starts WCS always from flashphoner user, if the user exists in system

2. The command

```
./webcallserver start
```

starts WCS from root when executing from root

```
sudo ./webcallserver start
```

or from flashphoner user, when executing from other non-root user

This affects the stanadlone mode too

```
./webcallserver start standalone
```

Launching build 5.2.976 and newer

Since build5.2.976, , the permissions to launch WSC are defined by the following parameter in `/usr/local/FlashphonerWebCallServer/bin/setenv.sh`file only:

On this value (default)

```
WCS_NON_ROOT=true
```

WCS is starting from flashphoner user

On this value

```
WCS_NON_ROOT=false
```

WCS is starting from root user

In this case, service can be started from root, user permissions to launch Java will be changed automatically.

Switching launch mode

Since build 5.2.1255 the following command is available to switch launch mode:

- switching to `root` mode

```
sudo ./webcallserver set-root-mode enable
```

- switching to `flashphoner` mode

```
sudo ./webcallserver set-root-mode disable
```

WCS will be stopped before settings changing and will be automatically started after settings changing to apply them.

Folder permissions setting when starting from flashphoner user

Since build5.2.976, write permissions to the server folders including custom folder are checked while starting WCS from flashphoner user. If permissions are not enough, WCS will not start with the following message in`/usr/local/FlashphonerWebCallServer/logs/startup.log` file

```
FlashphonerWebCallServer cannot be started from user flashphoner, please fix the permissions to the folders or
run 'webcallserver set-permissions'!
```

In this case, the following command should be executed

```
sudo ./webcallserver set-permissions
```

JVM parameters

Parameters are set in the wcs-core.properties file.

Additional launching options can be set in bin/setenv.sh file using the following varaibles:

WCS_JAVA_OPTS- the list of options for WCS Core

JVM parameters are checked for compatibility with current Java version on startup. The error messages are written to /usr/local /FlashphonerWebCallServer/logs/startup.log file according to error message returned by Java if JVM cannot start with parameters specified.

Java version automatic detection and JVM parameters correction

Since build5.2.972, Java version is detected automatically, and JVM parameters are corrected when WCS is starting after JDK update. JVM launch parameters may also be corrected by the following command

```
cd /usr/local/FlashphonerWebCallServer/bin
sudo ./webcallserver set-java-opts
```

In this case, the parameters are corrected in the wcs-core.properties file, the previous settings are copied to a file with.backup extension and a sequence number, for example

```
[root@localhost ~]# ls -l /usr/local/FlashphonerWebCallServer/conf/wcs-core.properties.backup.*
-rw-r--r--. 1 flashphoner flashphoner 1614 Jun 23 10:15 /usr/local/FlashphonerWebCallServer/conf/wcs-core.
properties.backup.0
-rw-r--r--. 1 flashphoner flashphoner 1543 Jun 23 10:17 /usr/local/FlashphonerWebCallServer/conf/wcs-core.
properties.backup.1
```

Note that garbage collector (GC) is not changing automatically in this case, but its parameters can be changed (command line key names for example).

Automatic WCS health checking after launch

When WCS process is launched, `webcallserver` script cheks if it is healthy waiting for 200 OK response to a special query

```
GET http://localhost:8081/health-check HTTP/1.1
```

Since build 5.2.1084 it is possible to set a maximum number of health checking tries using the following command line key

```
sudo ./webcallserver start --health-timeout 10
```

By default, 10 tries will be done with 1 second timeout between them. The script waits 1 second for response on every try. Therefore, a maximum waiting time may reach up to 20 seconds by default (`10 * (1+1)`).

If WCS process is not responding to all the queries, or the response is not 200 OK, the following message will be written to launch log `startup.log` and to console

```
FlashphonerWebCallServer started, but is not healthy, please try to restart
```

This health checking may be disabled if necessary by setting a zero tries

```
sudo ./webcallserver start --health-timeout 0
```