Liferay 7.2 Community Edition GA1 (Cluster Configuration) Docker Compose Project

In June 2019 (about five months ago), Liferay 7.2 GA1 was released. The great news was the return of the cluster OOTB, **without installing any other jar**.

This repository contains a Docker Compose project that allows you to get within a few minutes a Liferay cluster composed of two working nodes.

This Docker Compose contains this services:

1. **lb-haproxy**: HA Proxy (version 2.0.7) as Load Balancer
2. **liferay-portal-node-1**: Liferay 7.2 GA1 (with cluster support) node 1
3. **liferay-portal-node-2**: Liferay 7.2 GA1 (with cluster support) node 2
4. **postgres**: PostgreSQL 10 database
5. **es-node-1** and **es-node-2**: Elasticsearch 6.8.1 Cluster nodes

As for the shared directory for the Liferay document library, I decided to use a shared dock volume instead of NSF.

Consider that this project is intended for cluster development and testing. The cluster configuration is the default one with JGroups's UDP multicast, but unicast and TCP are still available.

For more information about *Liferay Cluster* and *Configure Liferay Portal to Connect to your Elasticsearch Cluster* you can read Liferay Portal Clustering and Connect to your Elasticsearch Cluster on Liferay Developer Network.

The **liferay** directory contains the following items:

1. **OSGi configs** (inside osgi/configs directory)
    - ElasticsearchConfiguration.config: contains elastic cluster configuration
    - AdvancedFileSystemStoreConfiguration.cfg: contains the configuration of the document library
2. **Portal properties** (inside configs directory)
    - portal-ext.properties: contains common configurations for Liferay, such as database connection, cluster enabling, document library, etc.

The **haproxy** directory contains the following items:

1. **HA Proxy**
    - haproxy.cfg: It contains the configuration to expose an endpoint http which balances the two Liferay nodes.

The **elastic** directory contains the optional configuration files for customizing cluster configuration.

The **deploy** directory contains two directories, one for each liferay node, where inside it is possible to copy the bundles to be installed on the respective node.

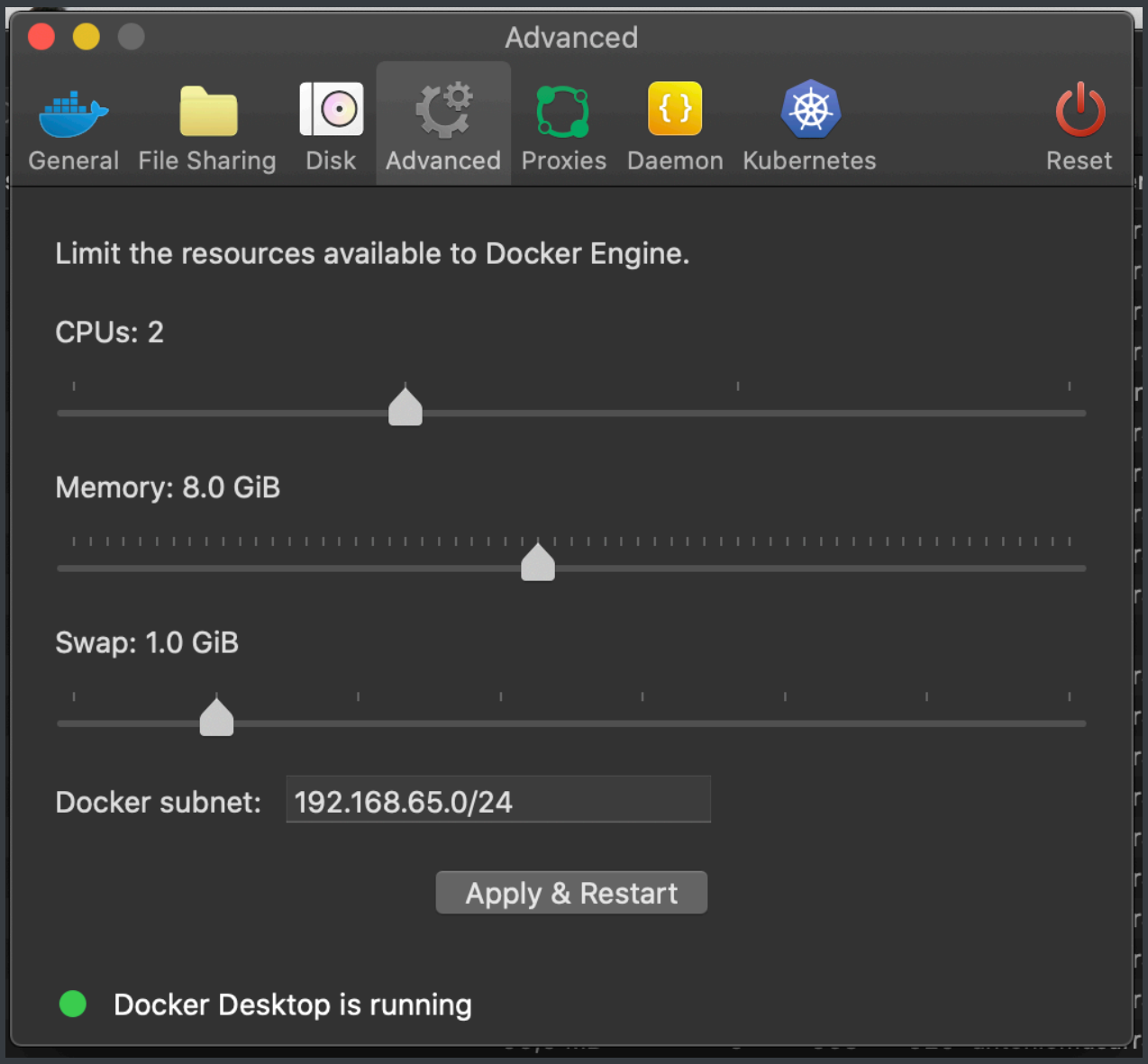
```
deploy
├── liferay-portal-node-1
└── liferay-portal-node-2
```

The fragment of the docker-compose.yml to highlight the bind type volumes dedicated to the deployment of bundles on Liferay nodes.

```
  liferay-portal-node-1:
    build:
      context: .
      dockerfile: Dockerfile-liferay
    hostname: liferay-portal-node-1.local
    volumes:
      - lfr-dl-volume:/data/liferay/document_library
      - ./deploy/liferay-portal-node-1:/etc/liferay/mount/deploy
  liferay-portal-node-2:
    build:
      context: .
      dockerfile: Dockerfile-liferay
    hostname: liferay-portal-node-2.local
    volumes:
      - lfr-dl-volume:/data/liferay/document_library
      - ./deploy/liferay-portal-node-2:/etc/liferay/mount/deploy
```

1. Usage

Install docker-compose and run docker with at least 8 GB of memory. The figure below shows the configuration of my Docker installation on macOS Catalina (v10.15).



To start a services from this Docker Compose, please run following `docker-compose` command, which will start a Liferay Portal 7.2 GA1 with cluster support running on Tomcat 9.0.17:

For the first start, proceed as follows:

```
$ docker-compose up -d liferay-portal-node-1
```

You can view output from containers following `docker-compose logs` or `docker-compose logs -f` for follow log output.

After the first Liferay node is on (liferay-portal-node-1), then run:

```
$ docker-compose up -d liferay-portal-node-2
```

After the two Liferay nodes are up, then pull the HA Proxy:

```
$ docker-compose up -d lb-haproxy
```

For the next start, you can run the only command:

```
$ docker-compose up -d
```

If you encounter (ERROR: An HTTP request took too long to complete) this issue regularly because of slow network conditions, consider setting COMPOSE_HTTP_TIMEOUT to a higher value (current value: 60).

```
$ COMPOSE_HTTP_TIMEOUT=200 docker-compose up -d
```

1.1 Check Liferay and Elasticsearch services

To check the successful installation of cluster bundles, you can connect to the Gogo Shell of each node (via telnet) and run the command:

The telnet ports of the GogoShell exposed by the two nodes are respectively: 21311 and 31311

```
g! lb -s multiple
```

and you should get the following output. Be careful that the three bundle must be in the active state.

```
START LEVEL 20
   ID|State      |Level|Symbolic name
  455|Active     |   10|com.liferay.portal.cache.multiple (3.0.5)|3.0.5
  461|Active     |   10|com.liferay.portal.scheduler.multiple (3.0.2)|3.0.2
  890|Active     |   10|com.liferay.portal.cluster.multiple (3.0.6)|3.0.6
```

On the logs of every Liferay instance you should see logs similar to those shown below.

```
liferay-portal-node-1_1  | 2019-11-05 23:38:56.039 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsClusterChannelFactory:158] Autodetecting JGroups outgoing IP
address and interface for www.google.com:80
```

```
liferay-portal-node-1_1  | 2019-11-05 23:38:56.234 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsClusterChannelFactory:197] Setting JGroups outgoing IP address to
172.21.0.5 and interface to eth0
liferay-portal-node-1_1  |
liferay-portal-node-1_1  | ----------------------------------------------
------------------
liferay-portal-node-1_1  | GMS: address=liferay-portal-node-1-51531,
cluster=liferay-channel-control, physical address=172.21.0.5:34684
liferay-portal-node-1_1  | ----------------------------------------------
------------------
liferay-portal-node-2_1  | 2019-11-05 23:38:56.908 INFO  [Incoming-
2,liferay-channel-control,liferay-portal-node-2-29145][JGroupsReceiver:91]
Accepted view [liferay-portal-node-2-29145|3] (2) [liferay-portal-node-2-
29145, liferay-portal-node-1-51531]
liferay-portal-node-1_1  | 2019-11-05 23:38:56.953 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsReceiver:91] Accepted view [liferay-portal-node-2-29145|3] (2)
[liferay-portal-node-2-29145, liferay-portal-node-1-51531]
liferay-portal-node-1_1  | 2019-11-05 23:38:56.957 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsClusterChannel:105] Create a new JGroups channel {channelName:
liferay-channel-control, localAddress: liferay-portal-node-1-51531,
properties: UDP(...)}
liferay-portal-node-1_1  |
liferay-portal-node-1_1  | ----------------------------------------------
------------------
liferay-portal-node-1_1  | GMS: address=liferay-portal-node-1-63028,
cluster=liferay-channel-transport-0, physical address=172.21.0.5:52103
liferay-portal-node-1_1  | ----------------------------------------------
------------------
liferay-portal-node-2_1  | 2019-11-05 23:38:57.374 INFO  [Incoming-
1,liferay-channel-transport-0,liferay-portal-node-2-44641]
[JGroupsReceiver:91] Accepted view [liferay-portal-node-2-44641|3] (2)
[liferay-portal-node-2-44641, liferay-portal-node-1-63028]
liferay-portal-node-1_1  | 2019-11-05 23:38:57.384 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsReceiver:91] Accepted view [liferay-portal-node-2-44641|3] (2)
[liferay-portal-node-2-44641, liferay-portal-node-1-63028]
liferay-portal-node-1_1  | 2019-11-05 23:38:57.389 INFO  [Start Level:
Equinox Container: 1ad0069e-0f93-4499-aac6-bc7da430eb11]
[JGroupsClusterChannel:105] Create a new JGroups channel {channelName:
liferay-channel-transport-0, localAddress: liferay-portal-node-1-63028,
properties: UDP(..)}
```

From this log we see the join between the two cluster Liferay nodes (liferay-portal-node-1 and liferay-portal-node-2).

To check the correct installation of the Elasticsearch cluster, just check the status of the cluster and the presence of the Liferay indexes. We can use REST services to get this information:

1. General info on installation of the Elasticsearch
2. Cluster Status
3. Indexes Status

```
$ curl http://localhost:9200
```

In output you should get the general info of the Elasticsearch.

```
{
  "name" : "ExxDLIz",
  "cluster_name" : "docker-elasticsearch",
  "cluster_uuid" : "owMQWJH9SRq3e9u3WE00pw",
  "version" : {
    "number" : "6.8.1",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "1fad4e1",
    "build_date" : "2019-06-18T13:16:52.517138Z",
    "build_snapshot" : false,
    "lucene_version" : "7.7.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

```
$ curl "http://localhost:9200/_cluster/health?pretty"
```

In output you should get the status of the cluster in green and the presence of two nodes.

```
{
  "cluster_name" : "docker-elasticsearch",
```

```
  "status" : "green",
  "timed_out" : false,
  "number_of_nodes" : 2,
  "number_of_data_nodes" : 2,
  "active_primary_shards" : 4,
  "active_shards" : 5,
  "relocating_shards" : 0,
  "initializing_shards" : 0,
  "unassigned_shards" : 0,
  "delayed_unassigned_shards" : 0,
  "number_of_pending_tasks" : 0,
  "number_of_in_flight_fetch" : 0,
  "task_max_waiting_in_queue_millis" : 0,
  "active_shards_percent_as_number" : 100.0
}
```

```
$ curl http://localhost:9200/_cat/indices
```

In output you should also get Liferay indices.

```
green open liferay-20101              crHED_E6Qi6gkaampSf4XA 1 0  86  2
123.4kb 123.4kb
green open liferay-20099              0wMuMUAkRH6kzcNmlraJqg 1 0   0  0
262b    262b
green open liferay-0                  mRcT7AGVRSqVwHvBz1wHcg 1 0 183 14
945.6kb 945.6kb
green open .monitoring-es-6-2019.10.23 IY6EW_hDR0m8Gc5GmZUVLA 1 1 282 20
713.8kb 466.3kb
```

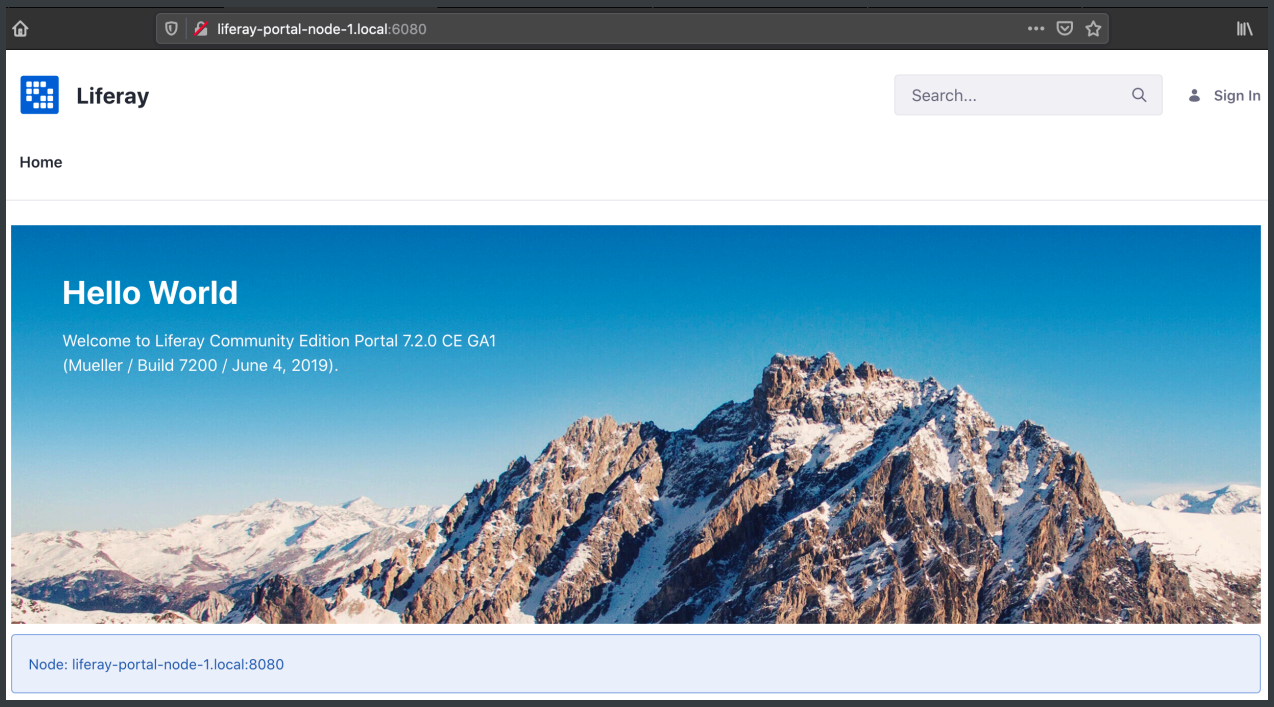
You can check if all the services have gone up using the `docker-compose ps` command.

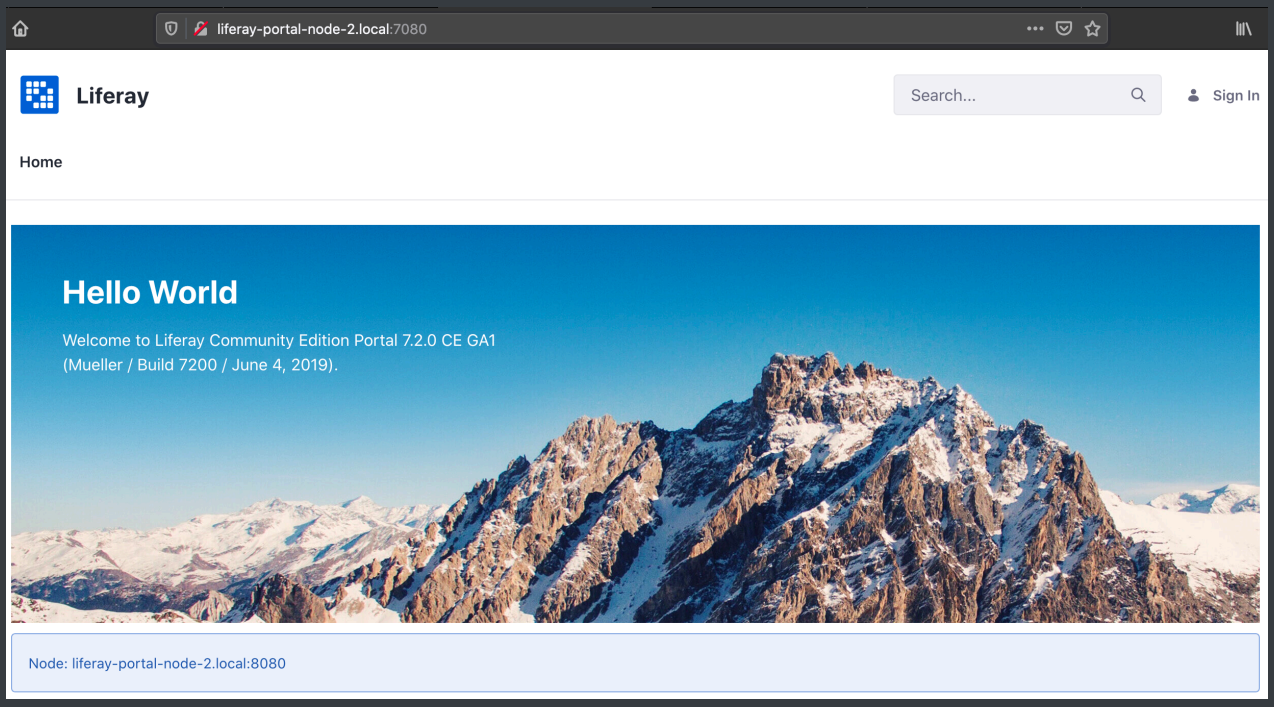
```
~/Progetti/MyProjects/Liferay/sources/docker-liferay-portal  7.2.0-ce-ga1-tomcat-postgres-cluster ●  docker-compose ps
WARNING: Some services (es-node-1, es-node-2) use the 'deploy' key, which will be ignored. Compose does not support 'deploy' configuration - use `docker stack
 deploy` to deploy to a swarm.
                Name                              Command               State                                    Ports
-------------------------------------------------------------------------------------------------------------------------------------------------------
docker-liferay-portal_es-node-1_1              /usr/local/bin/docker-entr ...   Up             0.0.0.0:9200->9200/tcp, 9300/tcp
docker-liferay-portal_es-node-2_1              /usr/local/bin/docker-entr ...   Up             9200/tcp, 9300/tcp
docker-liferay-portal_lb-haproxy_1             /docker-entrypoint.sh hapr ...   Up             0.0.0.0:80->80/tcp, 0.0.0.0:8181->8181/tcp
docker-liferay-portal_liferay-portal-node-1_1  /bin/sh -c /usr/local/bin/ ...   Up (healthy)   0.0.0.0:21311->11311/tcp, 0.0.0.0:6001->8000/tcp, 8009/tcp,
                                                                                              0.0.0.0:6080->8080/tcp
docker-liferay-portal_liferay-portal-node-2_1  /bin/sh -c /usr/local/bin/ ...   Up (healthy)   0.0.0.0:31311->11311/tcp, 0.0.0.0:7001->8000/tcp, 8009/tcp,
                                                                                              0.0.0.0:7080->8080/tcp
docker-liferay-portal_postgres_1               docker-entrypoint.sh postgres    Up             5432/tcp
```

After all the services are up, you can reach Liferay this way:

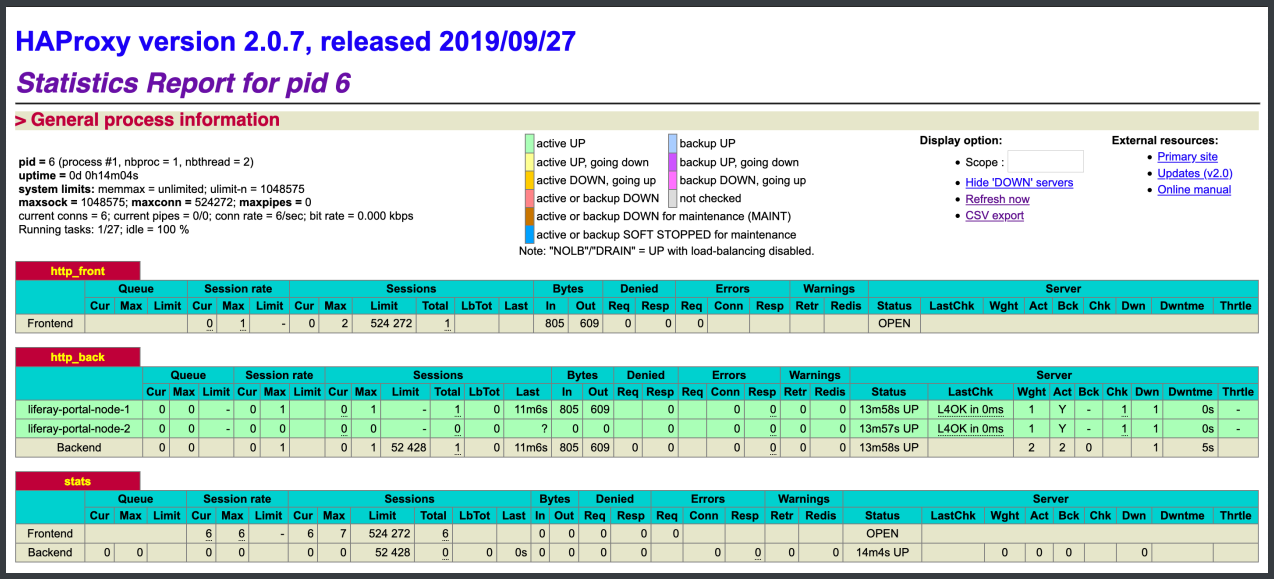
1. Via HA Proxy or Load Balancer at URL http://localhost
2. Accessing directly to the nodes:
    - Liferay Node 1: http://localhost:6080

- Liferay Node 2: http://localhost:7080





You can access the HA Proxy statistics report in this way: http://localhost:8181 (username/password: liferay/liferay)



HAProxy version 2.0.7, released 2019/09/27

Statistics Report for pid 6

> General process information

pid = 6 (process #1, nbproc = 1, nbthread = 2)
uptime = 0d 0h14m04s
system limits: memmax = unlimited; ulimit-n = 1048575
maxsock = 1048575; **maxconn =** 524272; **maxpipes =** 0
current conns = 6; current pipes = 0/0; conn rate = 6/sec; bit rate = 0.000 kbps
Running tasks: 1/27; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
backup UP
backup UP, going down
backup DOWN, going up
not checked
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide 'DOWN' servers
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v2.0)
- Online manual

http_front

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	1	-	0	2	524 272	1			805	609	0	0	0					OPEN								

http_back

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
liferay-portal-node-1	0	0	-	0	1		0	1	-	1	0	11m6s	805	609		0		0	0	0	0	13m58s UP	L4OK in 0ms	1	Y	-	1	1	0s	-
liferay-portal-node-2	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	13m57s UP	L4OK in 0ms	1	Y	-	1	1	0s	-
Backend	0	0		0	1		0	1	52 428	1	0	11m6s	805	609	0	0		0	0	0	0	13m58s UP		2	2	0		1	5s	

stats

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				6	6	-	6	7	524 272	6			0	0	0	0	0					OPEN								
Backend	0	0		0	0		0	0	52 428	0	0	0s	0	0	0	0		0	0	0	0	14m4s UP		0	0	0		0		

In my case, I inserted the following entries on my /etc/hosts file:

```
##
# Liferay 7.1 CE GA1 Cluster
##
127.0.0.1 liferay-portal-node-1.local
127.0.0.1 liferay-portal-node-2.local
127.0.0.1 liferay-portal.local
127.0.0.1 liferay-lb.local
```

To access Liferay through HA Proxy goto your browser at http://liferay-lb.local

License