```
In [1]: from genefab import GLDSCollection, MicroarrayExperiment
```

GLDSCollection() searches and stores up to **maxcount** GLDS datasets based on regular expressions passed to the arguments **ptype**, **organism**, **factor**, and **assay**.
The default value of **maxcount** is 25.
If either of the arguments is not passed, it is assumed to be a wildcard.

```
In [2]: collection = GLDSCollection(ptype="flight", organism="mus", factor="radiation", assay="transcript", maxcount=10)
        print(len(collection))
```

```
looking up ptype(s): "Spaceflight Study", "Spaceflight Project", "Spaceflight", "Flight Study", "Flight",
looking up organism(s): "Mus musculus",
looking up factor(s): "Absorbed Radiation Dose", "Ionizing Radiation", "Ionzing Radiation", "Irradiation", "post radiat
ion timepoint", "Radiation", "Radiation Distance", "Radiation dosage", "radiation dose", "radiation type", "Radiation,
Ionzing",
looking up assay(s): "transcription profiling",
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/search/?term=GLDS&type=cgene&size=10&ffield=Project+Type&f
value=Spaceflight+Study&ffield=Project+Type&fvalue=Spaceflight+Project&ffield=Project+Type&fvalue=Spaceflight&ffield=Pr
oject+Type&fvalue=Flight+Study&ffield=Project+Type&fvalue=Flight&ffield=organism&fvalue=Mus+musculus&ffield=Study+Facto
r+Name&fvalue=Absorbed+Radiation+Dose&ffield=Study+Factor+Name&fvalue=Ionizing+Radiation&ffield=Study+Factor+Name&fvalu
e=Ionzing+Radiation&ffield=Study+Factor+Name&fvalue=Irradiation&ffield=Study+Factor+Name&fvalue=post+radiation+timepoin
t&ffield=Study+Factor+Name&fvalue=Radiation&ffield=Study+Factor+Name&fvalue=Radiation+Distance&ffield=Study+Factor+Name
&fvalue=Radiation+dosage&ffield=Study+Factor+Name&fvalue=radiation+dose&ffield=Study+Factor+Name&fvalue=radiation+type&
ffield=Study+Factor+Name&fvalue=Radiation%2C+Ionzing&ffield=Study+Assay+Measurement+Type&fvalue=transcription+profiling
```

```
9
```

GLDS objects have multiple attributes. Here is an example of finding the datasets that contain raw array files (CELs)

```
In [3]: array_datasets = [
            glds for glds in collection
            if glds.has_raw_arrays
        ]
        print(len(array_datasets))
```

```
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-173/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-87/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-25/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-21/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-50/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-135/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-116/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-4/
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/data/GLDS-111/
```

```
8
```

```
In [4]: glds = array_datasets[0]
        print(glds.accession)
```

```
GLDS-87
```

Here we build a **MicroarrayExperiment** object from an existing **GLDS** object.
This triggers listing of archived files in the **GLDS** objects, and if they have not been downloaded before, they are fetched from the server. The next time this **GLDS** object is initialized, it will reuse the existing local files.
Similarly, as the unarchived versions of the data files are needed for further analyses, the unpacking is triggered. If the files have been unpacked before, this action will be skipped and existing files will be reused.

In [5]:
```
experiment = MicroarrayExperiment(glds)
experiment.annotation
```

```
Parsing url:  https://genelab-data.ndc.nasa.gov/genelab/data/study/filelistings/59c440a440eb233adce5aa5f
Downloading GLDS-87_metadata_Zanello_STS135-ISA.zip: 100%|██████████| 5/5 [00:00<?, ?KB/s]
Downloading GLDS-87_microarray_14R_(Mouse430_2).CEL.gz: 100%|██████████| 6226/6226 [00:54<00:00, 114.21KB/s]
Downloading GLDS-87_microarray_16R_(Mouse430_2).CEL.gz: 100%|██████████| 6128/6128 [00:57<00:00, 105.83KB/s]
Downloading GLDS-87_microarray_20R_(Mouse430_2).CEL.gz: 100%|██████████| 6235/6235 [00:21<00:00, 286.75KB/s]
Downloading GLDS-87_microarray_52R_(Mouse430_2).CEL.gz: 100%|██████████| 6150/6150 [00:28<00:00, 219.35KB/s]
Downloading GLDS-87_microarray_54R_(Mouse430_2).CEL.gz: 100%|██████████| 5877/5877 [00:35<00:00, 167.02KB/s]
Downloading GLDS-87_microarray_58R_(Mouse430_2).CEL.gz: 100%|██████████| 6182/6182 [00:45<00:00, 136.63KB/s]
Unpacking top-level files: 100%|██████████| 7/7 [00:02<00:00,  2.55file/s]
Unpacking second-level files: 100%|██████████| 6/6 [00:00<00:00, 6001.87file/s]
```

Out[5]:

	Spaceflight	**filename**
Sample Name		
16R	Ground	.genefab\MicroarrayExperiment_source\GLDS-87\G...
18R	Ground	.genefab\MicroarrayExperiment_source\GLDS-87\G...
20R	Ground	.genefab\MicroarrayExperiment_source\GLDS-87\G...
52R	Flight	.genefab\MicroarrayExperiment_source\GLDS-87\G...
54R	Flight	.genefab\MicroarrayExperiment_source\GLDS-87\G...
58R	Flight	.genefab\MicroarrayExperiment_source\GLDS-87\G...

In [10]:
```
experiment.factors
```

Out[10]:
```
{'Spaceflight': {'Flight', 'Ground'}}
```

Finally, we run limma. It picks up information from the annotation dataframe and builds the design matrix and the affymetrix dataset from it.
This part of the code is only a proof of concept, and a proper universal protocol needs to be decided upon and implemented. As you can see, the adjusted p-values that we get from running this proof-of-concept method are laughable.

```
In [9]: deg = experiment.limma(factor_name="Spaceflight")
        deg.sort_values(by="adj.P.Val")[:10]
```

```
Running Rscript 'C:\Users\Kirill\AppData\Local\Temp\tmpdnai0p2f' and storing to 'C:\Users\Kirill\AppData\Local\Temp\tmp
ez42k2kh'
```

Out[9]:

	logFC	AveExpr	t	P.Value	adj.P.Val	B
1430295_at	-0.916608	7.031751	-7.792632	0.000142	0.999328	-3.201115
1447746_at	0.036988	3.854953	0.434572	0.677654	0.999328	-4.798819
1417661_at	-0.041838	8.467491	-0.434558	0.677663	0.999328	-4.798822
1423652_at	-0.075915	7.292880	-0.434553	0.677666	0.999328	-4.798823
1425908_at	-0.052614	4.314412	-0.434495	0.677707	0.999328	-4.798834
1448905_at	0.072962	8.035464	0.434464	0.677728	0.999328	-4.798839
1427325_s_at	-0.087824	6.045436	-0.434455	0.677734	0.999328	-4.798841
1433639_at	0.062613	8.400961	0.434443	0.677743	0.999328	-4.798843
1447102_at	-0.048139	5.325603	-0.434420	0.677758	0.999328	-4.798847
1428635_at	-0.087399	7.126671	-0.434360	0.677800	0.999328	-4.798859