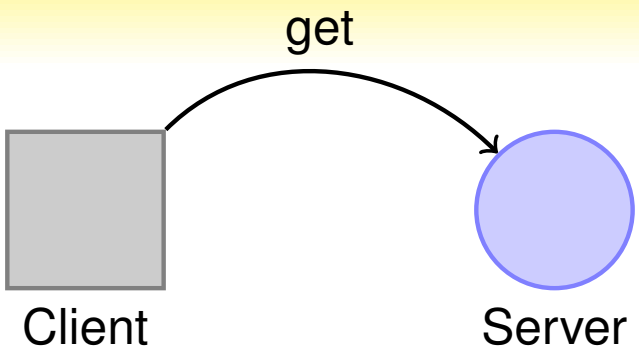
Exploring Biological Databases

Holger Dinkel

EMBO Practical Course Computational analysis of protein-protein interactions: From sequences to networks

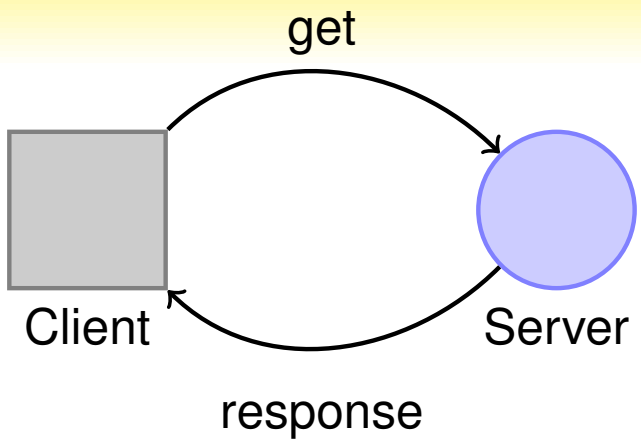
Web Services

Web Services



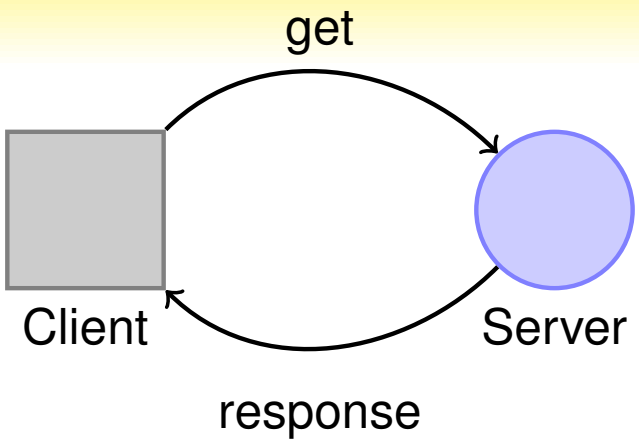
get: `http://www.uniprot.org/uniprot/P12931`

Web Services



get: `http://www.uniprot.org/uniprot/P12931`

response: **HTML**



get: `http://www.uniprot.org/uniprot/P12931.txt`
response: **TEXT/TSV**

```
ID SRC_HUMAN Reviewed; 536 AA.
AC P12931; E1P5V4; Q76P87; Q86VB9; Q9H5A8;
DT 01-OCT-1989, integrated into UniProtKB/Swiss-Prot.
DT 23-JAN-2007, sequence version 3.
DT 03-SEP-2014, entry version 187.
DE RecName:  Full=Proto-oncogene tyrosine-protein kinase Src;
...
```

A RESTful application

is an application that exposes its state and functionality as a set of resources that the clients can manipulate and conforms to a certain set of principles:

- All resources are uniquely addressable, usually through URIs; other addressing can also be used, though.
- All resources can be manipulated through a constrained set of well-known actions, usually CRUD (create, read, update, delete), represented most often through the HTTP's POST, GET, PUT and DELETE; it can be a different set or a subset though - for example, some implementations limit that set to read and modify only (GET and PUT) for example
- The data for all resources is transferred through any of a constrained number of well-known representations, usually HTML, XML or JSON;
- The communication between the client and the application is performed over a *stateless* protocol that allows for multiple layered intermediaries that can reroute and cache the requests and response packets transparently for the client and the application.

A RESTful application

is an application that exposes its state and functionality as a set of resources that the clients can manipulate and conforms to a certain set of principles:

- All resources are uniquely addressable, usually through URIs; other addressing can also be used, though.
- All resources can be manipulated through a constrained set of well-known actions, usually CRUD (create, read, update, delete), represented most often through the HTTP's POST, GET, PUT and DELETE; it can be a different set or a subset though - for example, some implementations limit that set to read and modify only (GET and PUT) for example
- The data for all resources is transferred through any of a constrained number of well-known representations, usually HTML, XML or JSON;
- The communication between the client and the application is performed over a *stateless* protocol that allows for multiple layered intermediaries that can reroute and cache the requests and response packets transparently for the client and the application.

A RESTful application

is an application that exposes its state and functionality as a set of resources that the clients can manipulate and conforms to a certain set of principles:

- All resources are uniquely addressable, usually through URIs; other addressing can also be used, though.
- All resources can be manipulated through a constrained set of well-known actions, usually CRUD (create, read, update, delete), represented most often through the HTTP's POST, GET, PUT and DELETE; it can be a different set or a subset though - for example, some implementations limit that set to read and modify only (GET and PUT) for example
- The data for all resources is transferred through any of a constrained number of well-known representations, usually HTML, XML or JSON;
- The communication between the client and the application is performed over a *stateless* protocol that allows for multiple layered intermediaries that can reroute and cache the requests and response packets transparently for the client and the application.

A RESTful application

is an application that exposes its state and functionality as a set of resources that the clients can manipulate and conforms to a certain set of principles:

- All resources are uniquely addressable, usually through URIs; other addressing can also be used, though.
- All resources can be manipulated through a constrained set of well-known actions, usually CRUD (create, read, update, delete), represented most often through the HTTP's POST, GET, PUT and DELETE; it can be a different set or a subset though - for example, some implementations limit that set to read and modify only (GET and PUT) for example
- The data for all resources is transferred through any of a constrained number of well-known representations, usually HTML, XML or JSON;
- The communication between the client and the application is performed over a *stateless* protocol that allows for multiple layered intermediaries that can reroute and cache the requests and response packets transparently for the client and the application.

REST Methods

Method defines what you want to do (**GET**=retrieve, **POST**=create/update, **DELETE**=remove).
We'll be using just GET requests which can be thought of as read-only access.
POST/DELETE are used to modify data on a server.

Protocol usually HTTP or HTTPS (secure)

URL defines a path to a resource

Parameters additional arguments, filters etc. usually in the form *parameter* = *value*; the first parameter is separated from the url by '?' while subsequent ones use '&'.

Example: searching for the term 'EMBO':

https://startpage.com/do/search?query=EMBO&with_language=lang_de

REST Methods

Method defines what you want to do (**GET**=retrieve, **POST**=create/update, **DELETE**=remove).
We'll be using just GET requests which can be thought of as read-only access.
POST/DELETE are used to modify data on a server.

Protocol usually HTTP or HTTPS (secure)

URL defines a path to a resource

Parameters additional arguments, filters etc. usually in the form *parameter* = *value*; the first parameter is separated from the url by '?' while subsequent ones use '&'.

Example: searching for the term 'EMBO':

https://startpage.com/do/search?query=EMBO&with_language=lang_de

REST Methods

Method defines what you want to do (**GET**=retrieve, **POST**=create/update, **DELETE**=remove).
We'll be using just GET requests which can be thought of as read-only access.
POST/DELETE are used to modify data on a server.

Protocol usually HTTP or HTTPS (secure)

URL defines a path to a resource

Parameters additional arguments, filters etc. usually in the form *parameter* = *value*; the first parameter is separated from the url by '?' while subsequent ones use '&'.

Example: searching for the term 'EMBO':

https://**startpage.com/do/search**?query=EMBO&with_language=lang_de

REST METHODS

Method defines what you want to do (**GET**=retrieve, **POST**=create/update, **DELETE**=remove).
We'll be using just GET requests which can be thought of as read-only access.
POST/DELETE are used to modify data on a server.

Protocol usually HTTP or HTTPS (secure)

URL defines a path to a resource

Parameters additional arguments, filters etc. usually in the form *parameter* = *value*; the first parameter is separated from the url by '?' while subsequent ones use '&'.

Example: searching for the term 'EMBO':

https://startpage.com/do/search**?query=EMBO&with_language=lang_de**

Note:

For all these examples, any common browser can be used, however for proper 'programmatic' access tools such as 'curl' or 'wget' on the Linux/Mac commandline are much more efficient and can easily be incorporated into little scripts...

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Benefits

Easy requests The data can be requested with simple HTTP requests and returned in a variety of programatic and bioinformatical relevant formats such as JSON, XML, YAML and FASTA.

Easy debugging Debugging can be done in any browser. While some might not call this real programming, it surely is the first step towards programmatically querying resources.

Reproducable You can write all your queries into a simple script and repeat the same query later. Even just saving the URL as a bookmark in your browser helps!

Powerful Any data can be made available via a REST service.

Bandwidth An API allows programmatic access to some information if one does not want to download the entire dataset.

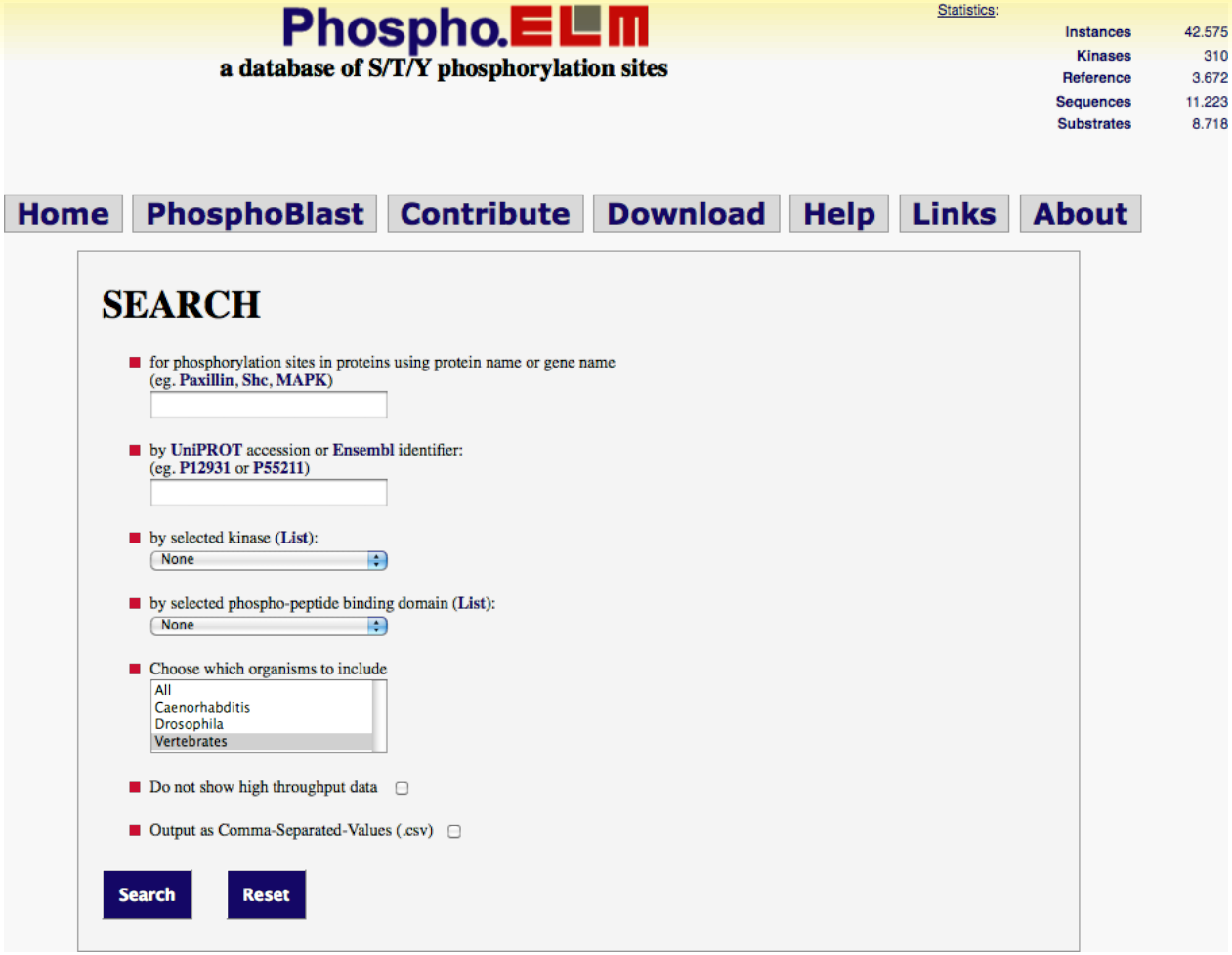
Standards By using existing protocols and best-methods (HTTP), all the existing knowledge can be reused (Caching, Redirecting, ...).

Widespread More and more resource providers change from fat/heavy webservices to this lightweight system, for obvious reasons.

Note:

Not meant to be a substitute for resources such as BioMART etc!

EXAMPLE: PHOSPHO.ELM



http://phospho.elm.eu.org/index.html

EXAMPLE: PHOSPHO.ELM

Access:

The PhosphoELM database can also be accessed via URL as follows:

- by ***substrate name:***
  http://phospho.elm.eu.org/bySubstrate/Paxillin.html
- by ***Uniprot ID:***
  http://phospho.elm.eu.org/byAccession/P12931.html
- by ***Uniprot ID*** and ***Position***
  http://phospho.elm.eu.org/byAccession/P12931/Pos17.html
- by ***ENSEMBL ID*** and multiple ***Positions***
  http://phospho.elm.eu.org/byAccession/ENSP00000265709/Pos216,231.html
- by ***Uniprot name:***
  http://phospho.elm.eu.org/byAccession/src_human.html
- by ***Kinase:***
  http://phospho.elm.eu.org/byKinase/Abl2.html
- by ***Binding domain:***
  http://phospho.elm.eu.org/byDomain/CBL_SH2.html
- retrieve a ***stored Sequence:***
  http://phospho.elm.eu.org/P12931.fasta
- retrieve data ***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931.csv
- retrieve data for a single position***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931/Pos12.csv
- retrieve data for ***multiple*** IDs ***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931,P55211.csv
- using ***web-services:***
  http://phospho.elm.eu.org/webservice/phosphoELMdb.wsdl

http://phospho.elm.eu.org/byAccession/P55211.**html**

EXAMPLE: PHOSPHO.ELM

Access:

The PhosphoELM database can also be accessed via URL as follows:

- by ***substrate name:***
  http://phospho.elm.eu.org/bySubstrate/Paxillin.html
- by ***Uniprot ID:***
  http://phospho.elm.eu.org/byAccession/P12931.html
- by ***Uniprot ID*** and ***Position***
  http://phospho.elm.eu.org/byAccession/P12931/Pos17.html
- by ***ENSEMBL ID*** and multiple ***Positions***
  http://phospho.elm.eu.org/byAccession/ENSP00000265709/Pos216,231.html
- by ***Uniprot name:***
  http://phospho.elm.eu.org/byAccession/src_human.html
- by ***Kinase:***
  http://phospho.elm.eu.org/byKinase/Abl2.html
- by ***Binding domain:***
  http://phospho.elm.eu.org/byDomain/CBL_SH2.html
- retrieve a ***stored Sequence:***
  http://phospho.elm.eu.org/P12931.fasta
- retrieve data ***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931.csv
- retrieve data for a single position***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931/Pos12.csv
- retrieve data for ***multiple*** IDs ***as CSV***
  http://phospho.elm.eu.org/byAccession/P12931,P55211.csv
- using ***web-services:***
  http://phospho.elm.eu.org/webservice/phosphoELMdb.wsdl

http://phospho.elm.eu.org/byAccession/P55211.**csv**

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/bySubstrate/cd66.html

Output:

Substrate: CD66 (Immunoglobulin)
Seq-ID: P13688 [*Homo sapiens*]
Interaction Network(s): NetworKIN
External Source(s): PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

Res.	Pos.	Sequence	Kinase	PMID	Src	Cons.	ELM	Binding Domain	SMART/Pfam	IUPRED score	PDB	P3D Acc.
Y	493	DPPNKMNEVTYSTLNFEAQQP	-	9867848	LTP	1.00		-	-	0.65	-	low
S	508	FEAQQPTQPTSASPSLTATEI	-	11850617	LTP	1.00		-	-	0.65	-	low
Y	520	SPSLTATEIIYSEVKKQ	-	9867848	LTP	1.00		-	-	0.38	-	low

Substrate: CD66 (Immunoglobulin)
Seq-ID: P31809 [*Mus musculus*]
Interaction Network(s): -
External Source(s): PHOSIDA PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/**bySubstrate**/cd66.html

- **Query by Substrate name**
- Substrate name
- Output as HTML

Output:

Substrate: CD66 (Immunoglobulin)
Seq-ID: P13688 [*Homo sapiens*]
Interaction Network(s): NetworKIN
External Source(s): PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

Res.	Pos.	Sequence	Kinase	PMID	Src	Cons.	ELM	Binding Domain	SMART/Pfam	IUPRED score	PDB	P3D Acc.
Y	493	DPPNKMNEVTYSTLNFEAQQP	-	9867848	LTP	1.00		-	-	0.65	-	low
S	508	FEAQQPTQPTSASPSLTATEI	-	11850617	LTP	1.00		-	-	0.65	-	low
Y	520	SPSLTATEIIYSEVKKQ	-	9867848	LTP	1.00		-	-	0.38	-	low

Substrate: CD66 (Immunoglobulin)
Seq-ID: P31809 [*Mus musculus*]
Interaction Network(s): -
External Source(s): PHOSIDA PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/bySubstrate/**cd66**.html

- Query by Substrate name
- **Substrate name**
- Output as HTML

Output:

Substrate: CD66 (Immunoglobulin)
Seq-ID: P13688 [*Homo sapiens*]
Interaction Network(s): NetworKIN
External Source(s): PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

Res.	Pos.	Sequence	Kinase	PMID	Src	Cons.	ELM	Binding Domain	SMART/Pfam	IUPRED score	PDB	P3D Acc.
Y	493	DPPNKMNEVTYSTLNFEAQQP	-	9867848	LTP	1.00		-	-	0.65	-	low
S	508	FEAQQPTQPTSASPSLTATEI	-	11850617	LTP	1.00		-	-	0.65	-	low
Y	520	SPSLTATEIIYSEVKKQ	-	9867848	LTP	1.00		-	-	0.38	-	low

Substrate: CD66 (Immunoglobulin)
Seq-ID: P31809 [*Mus musculus*]
Interaction Network(s): -
External Source(s): PHOSIDA PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/bySubstrate/cd66.**html**

- Query by Substrate name
- Substrate name
- **Output as HTML**

Output:

Substrate: CD66 (Immunoglobulin)
Seq-ID: P13688 [*Homo sapiens*]
Interaction Network(s): NetworKIN
External Source(s): PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

Res.	Pos.	Sequence	Kinase	PMID	Src	Cons.	ELM	Binding Domain	SMART/Pfam	IUPRED score	PDB	P3D Acc.
Y	493	DPPNKMNEVTYSTLNFEAQQP	-	9867848	LTP	1.00		-	-	0.65	-	low
S	508	FEAQQPTQPTSASPSLTATEI	-	11850617	LTP	1.00		-	-	0.65	-	low
Y	520	SPSLTATEIIYSEVKKQ	-	9867848	LTP	1.00		-	-	0.38	-	low

Substrate: CD66 (Immunoglobulin)
Seq-ID: P31809 [*Mus musculus*]
Interaction Network(s): -
External Source(s): PHOSIDA PhosphoSitePlus
MINT Interaction(s): -
GO-Terms: [show]
Conservation:

Click on table headers for sorting

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/byAccession/P12931/Pos12,17.csv

Output:

```
Acc.; Res.; Pos.; Context; Kinase; PMID; Source; ConScore; ELM; Domain; SMART; IUPRED; PDB; P3D-
P12931; S; 12; SNKSKPKDASQRRRSLEPAE; none; 2136766; 1; 0.21; ; -; ; 0.9168; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 18088087; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17192257; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17081983; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; PKA_group; 11804588; 1; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
...
```

Example: Phospho.ELM

Query

http://phospho.elm.eu.org/**byAccession**/P12931/Pos12,17.csv

- **query by Uniprot Accession**
- Protein Sequence Accession/ID
- Position / multiple Positions
- Output as CSV (character separated values)

Output:

```
Acc.; Res.; Pos.; Context; Kinase; PMID; Source; ConScore; ELM; Domain; SMART; IUPRED; PDB; P3D-
P12931; S; 12; SNKSKPKDASQRRRSLEPAE; none; 2136766; 1; 0.21; ; -; ; 0.9168; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 18088087; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17192257; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17081983; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; PKA_group; 11804588; 1; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
...
```

Example: Phospho.ELM

Query

http://phospho.elm.eu.org/byAccession/**P12931**/Pos12,17.csv

- query by Uniprot Accession
- **Protein Sequence Accession/ID**
- Position / multiple Positions
- Output as CSV (character separated values)

Output:

```
Acc.; Res.; Pos.; Context; Kinase; PMID; Source; ConScore; ELM; Domain; SMART; IUPRED; PDB; P3D-
P12931; S; 12; SNKSKPKDASQRRRSLEPAE; none; 2136766; 1; 0.21; ; -; ; 0.9168; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 18088087; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17192257; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17081983; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; PKA_group; 11804588; 1; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
...
```

EXAMPLE: PHOSPHO.ELM

Query

http://phospho.elm.eu.org/byAccession/P12931/**Pos12,17**.csv

- query by Uniprot Accession
- Protein Sequence Accession/ID
- **Position / multiple Positions**
- Output as CSV (character separated values)

Output:

```
Acc.; Res.; Pos.; Context; Kinase; PMID; Source; ConScore; ELM; Domain; SMART; IUPRED; PDB; P3D-
P12931; S; 12; SNKSKPKDASQRRRSLEPAE; none; 2136766; 1; 0.21; ; -; ; 0.9168; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 18088087; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17192257; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17081983; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; PKA_group; 11804588; 1; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
...
```

Example: Phospho.ELM

Query

http://phospho.elm.eu.org/byAccession/P12931/Pos12,17.csv

- query by Uniprot Accession
- Protein Sequence Accession/ID
- Position / multiple Positions
- **Output as CSV (character separated values)**

Output:

```
Acc.; Res.; Pos.; Context; Kinase; PMID; Source; ConScore; ELM; Domain; SMART; IUPRED; PDB; P3D-
P12931; S; 12; SNKSKPKDASQRRRSLEPAE; none; 2136766; 1; 0.21; ; -; ; 0.9168; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 18088087; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17192257; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; none; 17081983; 2; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
P12931; S; 17; PKDASQRRRSLEPAENVHGA; PKA_group; 11804588; 1; 0.24; MOD_PKA_1; -; ; 0.8828; -; ;
...
```

Example: ELM

Search ELM Instances

Full-Text Search (use "*" to get all instances)

P12931

Filter by instance Logic

Filter by organism

submit Reset

export 5 instances as: gff pir fasta tsv

5 Instances for search term 'P12931':

(click table headers for sorting; Notes column: =Number of Switches, =Number of Interactions)

ELM identifier	Acc., Gene-, Name	Start	End	Subsequence	Logic	#Ev.	Organism	Notes
LIG_SH2_SRC	P12931 SRC SRC_HUMAN	530	533	AFLEDYFTSTEPQYQPGENL	TP	1	Homo sapiens (Human)	1
LIG_SH3_4	P12931 SRC SRC_HUMAN	252	259	TVCPTSKPQTQGLAKDAWEI	TP	0	Homo sapiens (Human)	
MOD_CDK_1	P12931 SRC SRC_HUMAN	72	78	GFNSSDTVTSPQRAGPLAGG	TP	1	Homo sapiens (Human)	
MOD_NMyristoyl	P12931 SRC SRC_HUMAN	1	7	MGSNKSKPKDASQRRRSLEP	TP	0	Homo sapiens (Human)	
MOD_TYR_CSK	P12931 SRC SRC_HUMAN	526	534	AFLEDYFTSTEPQYQPGENL	TP	1	Homo sapiens (Human)	

Please cite: The Eukaryotic Linear Motif Resource ELM: 10 Years and Counting (PMID: 24214962)

feedback@elm.eu.org

ELM data can be downloaded & distributed for non-commercial use according to the ELM Software License Agreement

EXAMPLE: ELM

Search ELM Instances

Full-Text Search (use "*" to get all instances)

P12931

Filter by instance Logic

Filter by organism

submit Reset

export 5 instances as: gff pir fasta tsv

5 Instances for search term 'P12931':

(click table headers for sorting; Notes column: =Number of Switches, =Number of Interactions)

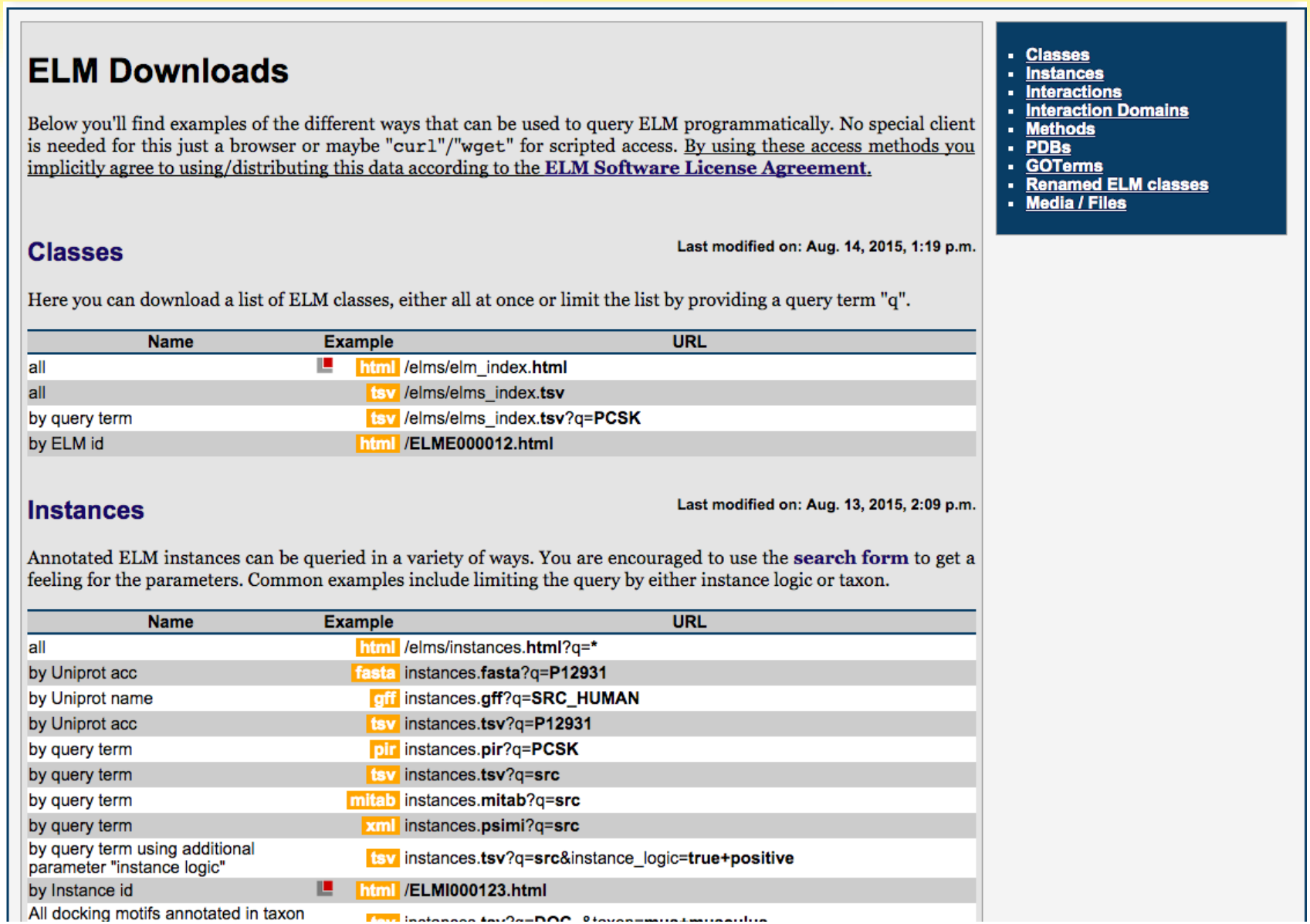
ELM identifier	Acc., Gene-, Name	Start	End	Subsequence	Logic	#Ev.	Organism	Notes
LIG_SH2_SRC	P12931 SRC SRC_HUMAN	530	533	AFLEDYFTSTEPQYQPGENL	TP	1	Homo sapiens (Human)	1
LIG_SH3_4	P12931 SRC SRC_HUMAN	252	259	TVCPTSKPQTQGLAKDAWEI	TP	0	Homo sapiens (Human)	
MOD_CDK_1	P12931 SRC SRC_HUMAN	72	78	GFNSSDTVTSPQRAGPLAGG	TP	1	Homo sapiens (Human)	
MOD_NMyristoyl	P12931 SRC SRC_HUMAN	1	7	MGSNKSKPKDASQRRRSLEP	TP	0	Homo sapiens (Human)	
MOD_TYR_CSK	P12931 SRC SRC_HUMAN	526	534	AFLEDYFTSTEPQYQPGENL	TP	1	Homo sapiens (Human)	

Please cite: The Eukaryotic Linear Motif Resource ELM: 10 Years and Counting (PMID: 24214962)

feedback@elm.eu.org

ELM data can be downloaded & distributed for non-commercial use according to the ELM Software License Agreement

EXAMPLE: ELM



ELM Downloads

Below you'll find examples of the different ways that can be used to query ELM programmatically. No special client is needed for this just a browser or maybe "curl"/"wget" for scripted access. By using these access methods you implicitly agree to using/distributing this data according to the **ELM Software License Agreement.**

- Classes
- Instances
- Interactions
- Interaction Domains
- Methods
- PDBs
- GOTerms
- Renamed ELM classes
- Media / Files

Classes

Last modified on: Aug. 14, 2015, 1:19 p.m.

Here you can download a list of ELM classes, either all at once or limit the list by providing a query term "q".

Name	Example	URL
all	html	/elms/elm_index.**html**
all	tsv	/elms/elms_index.**tsv**
by query term	tsv	/elms/elms_index.**tsv**?q=**PCSK**
by ELM id	html	**/ELME000012.html**

Instances

Last modified on: Aug. 13, 2015, 2:09 p.m.

Annotated ELM instances can be queried in a variety of ways. You are encouraged to use the **search form** to get a feeling for the parameters. Common examples include limiting the query by either instance logic or taxon.

Name	Example	URL
all	html	/elms/instances.**html**?q=*
by Uniprot acc	fasta	instances.**fasta**?q=**P12931**
by Uniprot name	gff	instances.**gff**?q=**SRC_HUMAN**
by Uniprot acc	tsv	instances.**tsv**?q=**P12931**
by query term	pir	instances.**pir**?q=**PCSK**
by query term	tsv	instances.**tsv**?q=**src**
by query term	mitab	instances.**mitab**?q=**src**
by query term	xml	instances.**psimi**?q=**src**
by query term using additional parameter "instance logic"	tsv	instances.**tsv**?q=**src**&instance_logic=**true+positive**
by Instance id	html	**/ELMI000123.html**
All docking motifs annotated in taxon	tsv	instances.**tsv**?q=**DOC_**&taxon=**mus+musculus**

EXAMPLE: ELM

ELM Downloads

Below you'll find examples of the different ways that can be used to query ELM programmatically. No special client is needed for this just a browser or maybe "curl"/"wget" for scripted access. By using these access methods you implicitly agree to using/distributing this data according to the **ELM Software License Agreement.**

- Classes
- Instances
- Interactions
- Interaction Domains
- Methods
- PDBs
- GOTerms
- Renamed ELM classes
- Media / Files

Classes

Last modified on: Aug. 14, 2015, 1:19 p.m.

Here you can download a list of ELM classes, either all at once or limit the list by providing a query term "q".

Name	Example	URL
all	html /elms/elm_index.**html**	
all	tsv /elms/elms_index.**tsv**	
by query term	tsv /elms/elms_index.**tsv**?q=**PCSK**	
by ELM id	html /**ELME000012.html**	

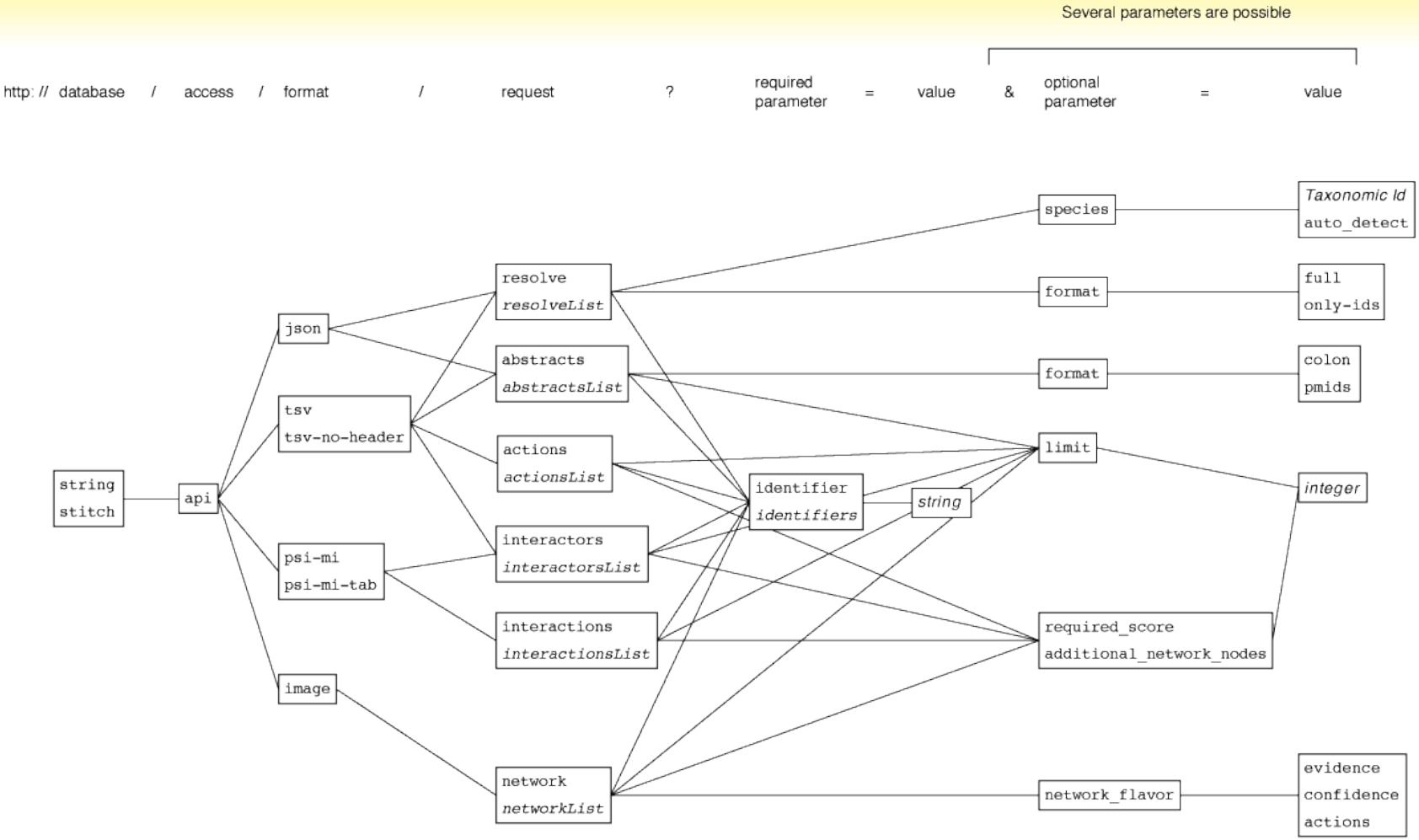
Instances

Last modified on: Aug. 13, 2015, 2:09 p.m.

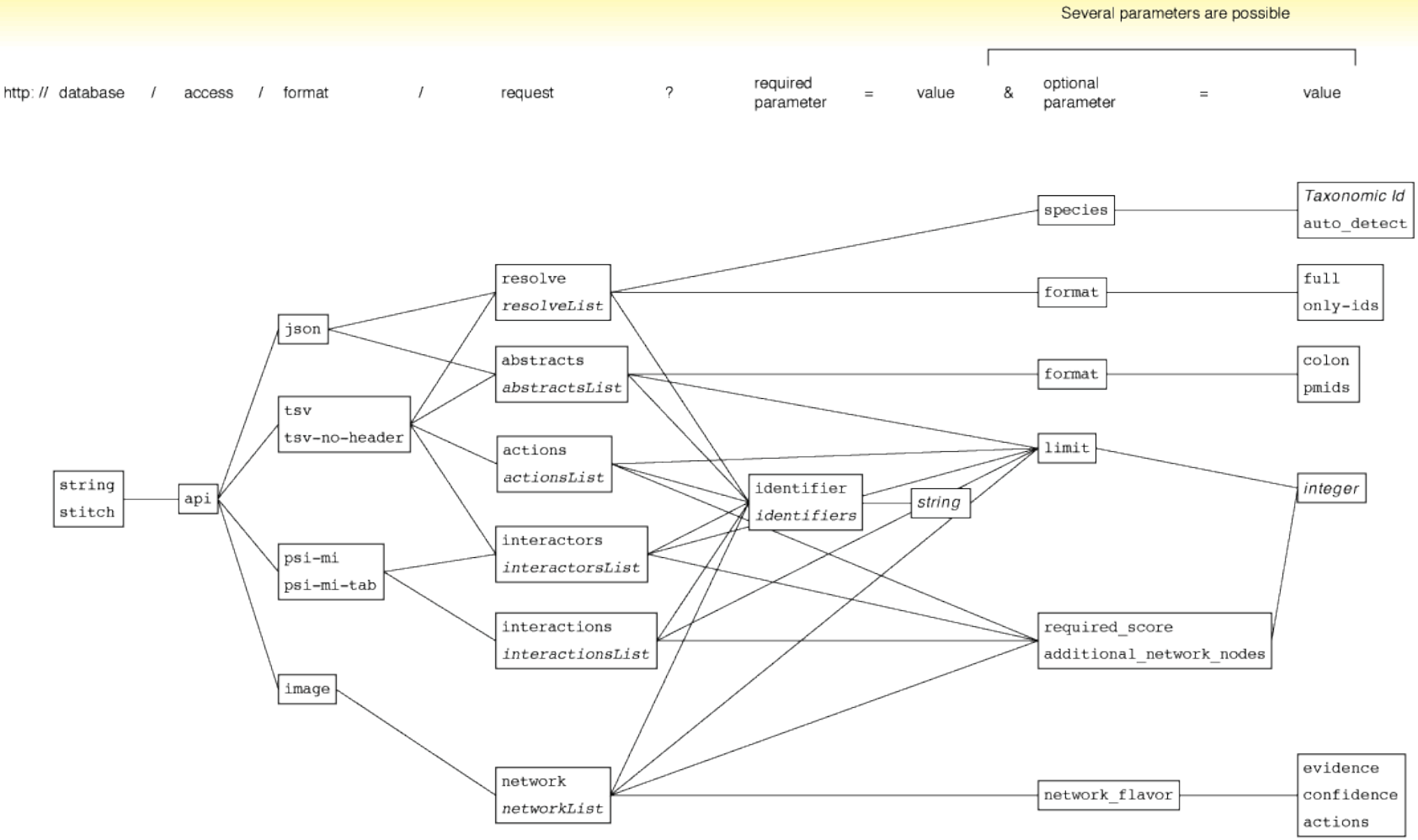
Annotated ELM instances can be queried in a variety of ways. You are encouraged to use the **search form** to get a feeling for the parameters. Common examples include limiting the query by either instance logic or taxon.

Name	Example	URL
all	html /elms/instances.**html**?q=*	
by Uniprot acc	fasta instances.**fasta**?q=**P12931**	
by Uniprot name	gff instances.**gff**?q=**SRC_HUMAN**	
by Uniprot acc	tsv instances.**tsv**?q=**P12931**	
by query term	pir instances.**pir**?q=**PCSK**	
by query term	tsv instances.**tsv**?q=**src**	
by query term	mitab instances.**mitab**?q=**src**	
by query term	xml instances.**psimi**?q=**src**	
by query term using additional parameter "instance logic"	tsv instances.**tsv**?q=**src**&instance_logic=**true+positive**	
by Instance id	html /**ELMI000123.html**	
All docking motifs annotated in taxon	tsv instances.tsv?q=DOC_&taxon=mus+musculus	

EXAMPLE: STRING / STITCH

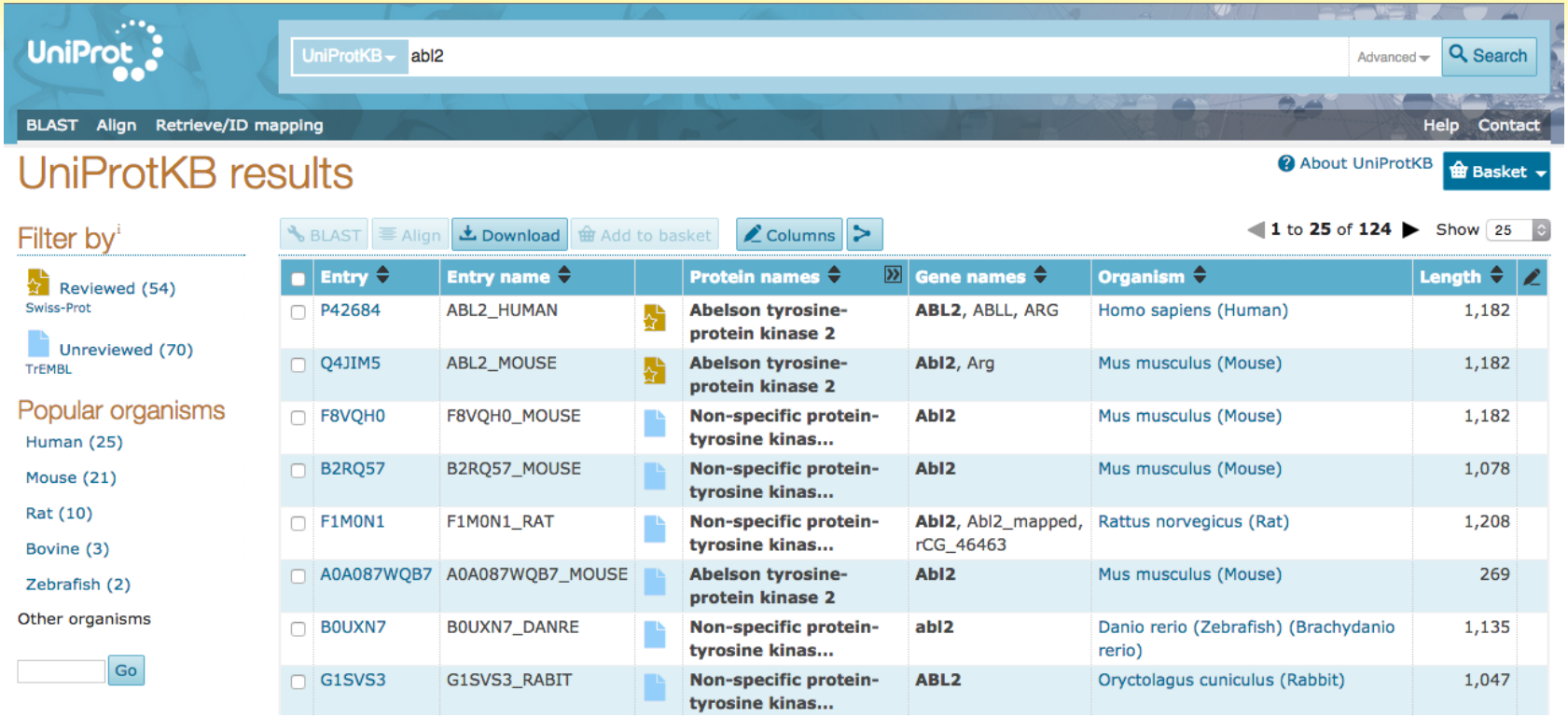


EXAMPLE: STRING / STITCH

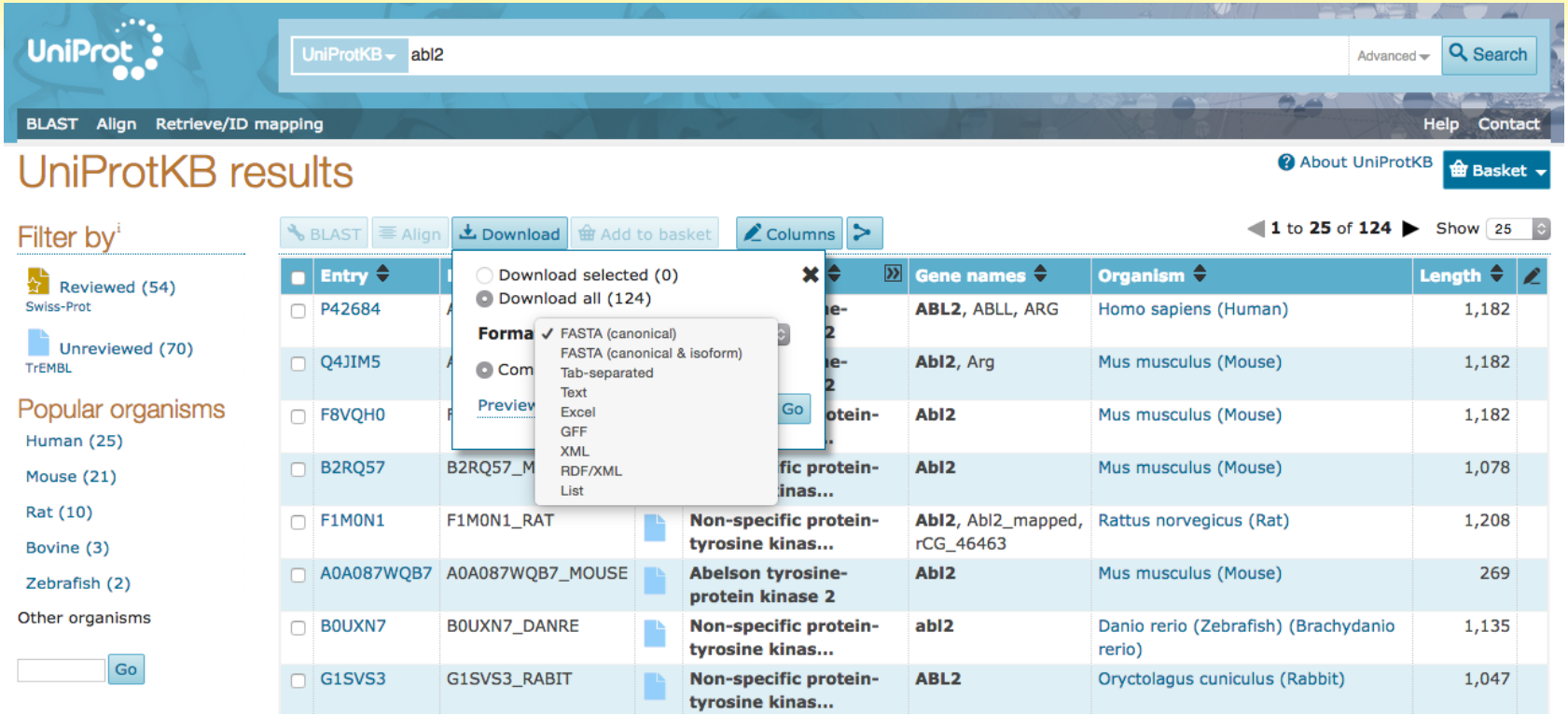


http://string-db.org/api/psi-mi-tab/interactions?identifier=YOL086C&additional_network_nodes=2

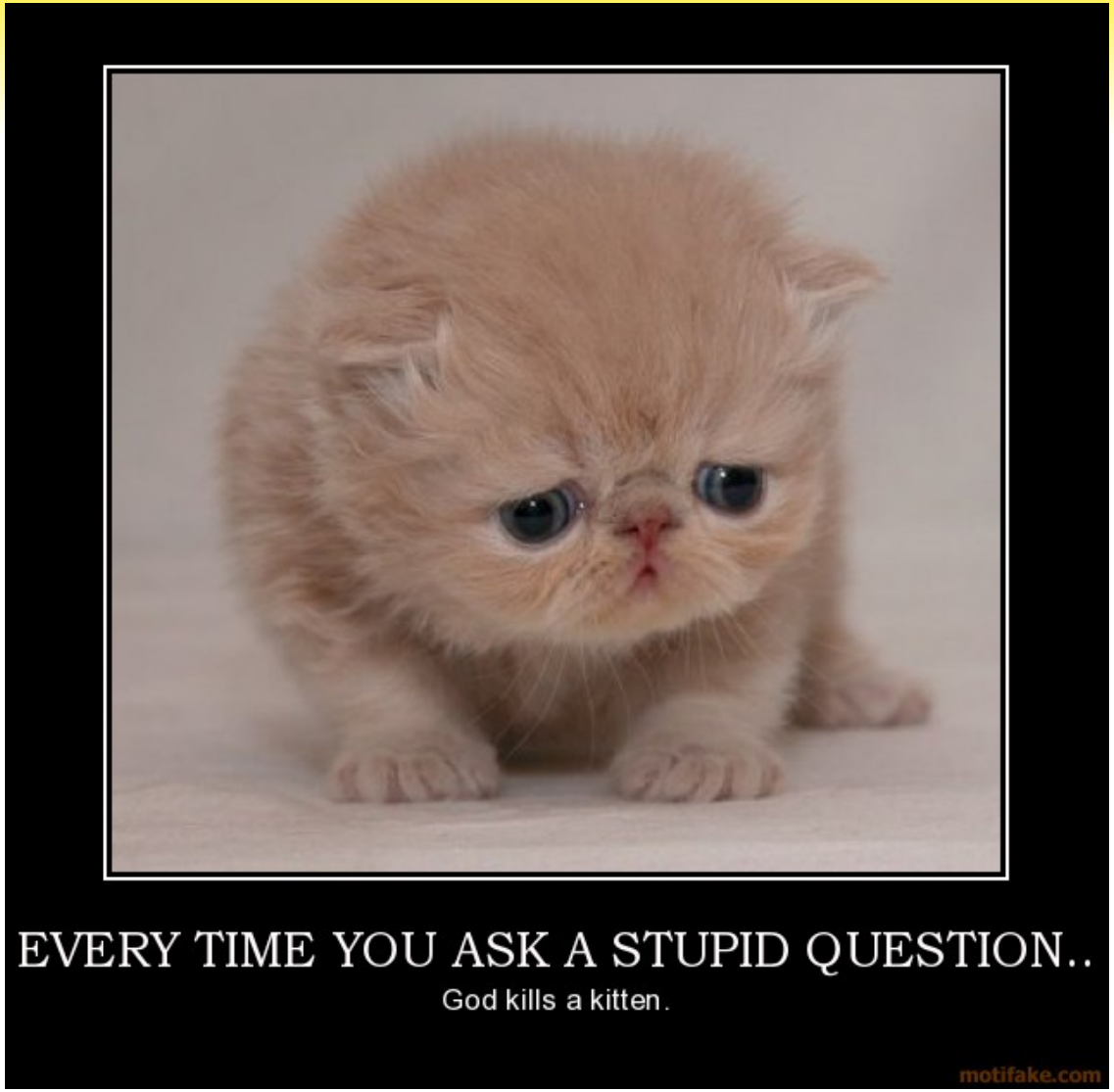
Example: UniProt



Example: UniProt



Questions?


EVERY TIME YOU ASK A STUPID QUESTION..
God kills a kitten.
motifake.com