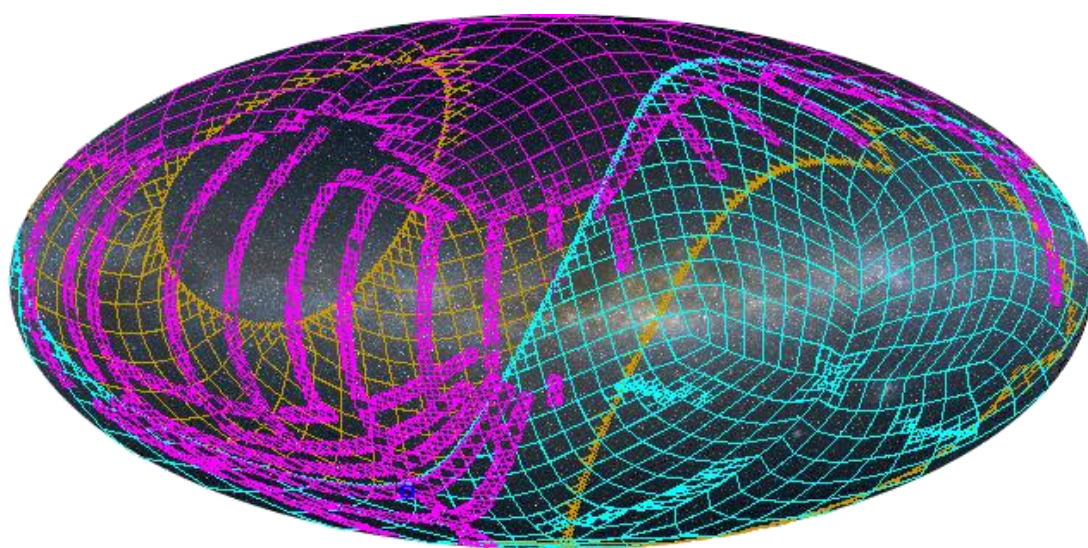


MocServer

Manuel de l'utilisateur

Dédié aux versions 5.06 et suivantes



MocServer – Manuel de l'utilisateur
Pierre Fernique

Mars 2023 (v2)

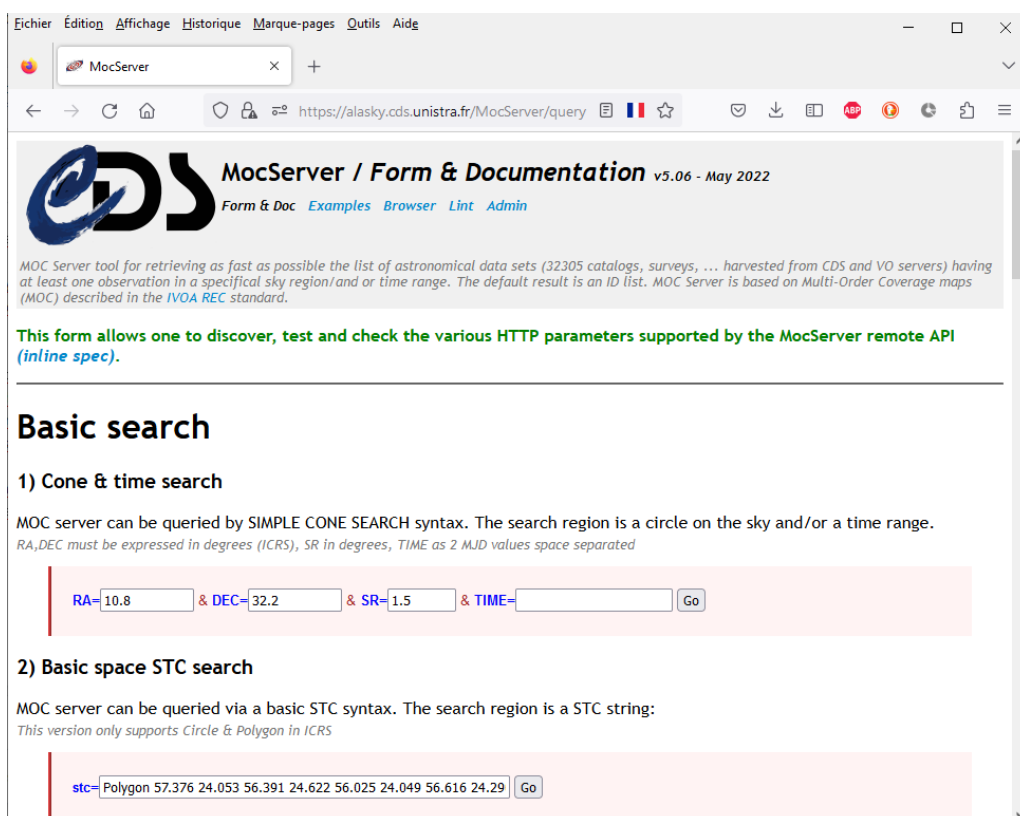
Centre de Données astronomiques de Strasbourg
Observatoire astronomique de Strasbourg

© 2023 – Université de Strasbourg / CNRS – sous Licence Ouverte (compatible CC-BY)

Introduction

Le MocServer est un service d'annuaire de ressources astronomiques. Il est basé sur les couvertures spatiales et/ou temporelles des collections de données qu'il gère. Son but principal est de fournir le plus rapidement possible (quelques millisecondes) la liste des ressources astronomiques concernées par une région particulière du ciel.

Le MocServer a été mise en place en 2015 par le Centre de Données astronomiques de Strasbourg (CDS). Il est utilisé principalement par les outils du CDS, tels Aladin Desktop, Aladin Lite, Simbad, le portail du CDS, etc. En 2023 il contient 32 000 entrées. Ce service est interrogeable à distance via une API HTTP. Une librairie python est également disponible sur Github au sein du package « astroquery »¹.



Fichier Édition Affichage Historique Marque-pages Outils Aide

MocServer

https://alasky.cds.unistra.fr/MocServer/query

MocServer / Form & Documentation v5.06 - May 2022

Form & Doc Examples Browser Lint Admin

MOC Server tool for retrieving as fast as possible the list of astronomical data sets (32305 catalogs, surveys, ... harvested from CDS and VO servers) having at least one observation in a specific sky region/and or time range. The default result is an ID list. MOC Server is based on Multi-Order Coverage maps (MOC) described in the *IVOA REC* standard.

This form allows one to discover, test and check the various HTTP parameters supported by the MocServer remote API (inline spec).

Basic search

1) Cone & time search

MOC server can be queried by SIMPLE CONE SEARCH syntax. The search region is a circle on the sky and/or a time range.
RA,DEC must be expressed in degrees (ICRS), SR in degrees, TIME as 2 MJD values space separated

RA=10.8 & DEC=32.2 & SR=1.5 & TIME= Go

2) Basic space STC search

MOC server can be queried via a basic STC syntax. The search region is a STC string:
This version only supports Circle & Polygon in ICRS

stc= Polygon 57.376 24.053 56.391 24.622 56.025 24.049 56.616 24.29 Go

¹ <https://astroquery.readthedocs.io/en/latest/cds/cds.html> (compatible MocServer 4.0, ne prend en compte que les MOC spatiaux et pas encore les MOC temporels)

Principe de fonctionnement

Le MocServer gère quelques milliers de descriptions de collections de données astronomiques. Chacune d'elle est repérée par un identifiant unique (ID), une liste de quelques propriétés (properties), et surtout la couverture associée (MOC). Cette couverture peut être soit spatiale, soit temporelle, soit les deux. Elle est décrite au moyen d'un « *Multi Order Coverage map* », ou MOC. Il s'agit d'une méthode standardisée par l'IVOA, décrite dans le document MOC 2.0², extrêmement efficace pour effectuer des opérations d'union, d'intersection et de comparaison. L'ensemble de ces données – les couples (properties, MOC) pour chaque ID – est chargée en RAM au démarrage du MocServer afin de lui permettre de répondre très rapidement aux requêtes.

Une requête typique consiste à demander au MocServer la liste des identificateurs (ID) des collections qui ont au moins une observation dans une zone du ciel. Le MocServer va procéder de la manière suivante :

1. Il détermine le MOC correspondant à la zone du ciel demandée ;
2. Il retient les collections dont l'intersection de son MOC avec celui de la zone est non nulle ;
3. Il retourne la liste des IDs de ces collections.

D'autre part, il est possible de filtrer ces résultats en ajoutant des contraintes sur différents champs des propriétés (description, provenance, copyright, etc).

Le résultat d'une telle requête peut ne pas se limiter aux identificateurs, mais s'étendre aux propriétés elles-mêmes fournissant un « service d'annuaire de méta données ».

Sites Web

Le MocServer est interrogeable à l'adresse ci-dessous. Il est disponible 24h/24 sans interruption :

<https://alasky.cds.unistra.fr/MocServer/query>

Un site miroir est également disponible. Il contient les mêmes descriptions de collections avec un décalage de mise à jour d'au plus 24 heures. Il est localisé à l'adresse suivante :

<https://alaskybis.cds.unistra.fr/MocServer/query>

Ces URLs sans paramètres additionnels retournent une page Web avec de multiples formulaires d'interrogations pour tester les différents paramètres d'interrogations possibles (cf. copie d'écran ci-dessus). C'est un moyen très efficace pour tester une interrogation, et en connaître la syntaxe, sans avoir à lire le présent manuel.

Le formulaire HTML du MocServer offre également quatre autres interfaces :

1. Une liste d'exemple d'utilisations
=> <https://.../MocServer/example>
2. Une page permettant la navigation directe dans les données des collections gérées
=> <https://.../MocServer/browse>
3. Un formulaire de vérification de la conformité d'un MOC avec le standard IVOA
=> <https://.../MocServer/lint>

² <https://www.ivoa.net/documents/MOC/>

4. Une page d'administration distante du MocServer (accès restreint)
=> <https://.../MocServer/admin>

Origines des données

Le MocServer contient actuellement plusieurs milliers de descriptions de collections de données. Elles proviennent principalement de 3 sources :

1. Les « HiPS lists » des serveurs HiPS ;
2. Une partie des collections décrites dans l'annuaire des données de l'IVOA (VO registry) ;
3. Une partie des collections de données spécifiques au CDS (VizieR et Simbad).

La mise à jour de ces données est faite quotidiennement. La liste des collections indexées répond au choix éditorial du CDS. Elle est fonction des besoins des outils clients visés.

Avertissement d'utilisation. Le MocServer est un outil complémentaire aux services à l'origine des données. Il s'agit d'un annuaire facilitant la sélection des collections indexés, suivants des critères de recherche principalement positionnels et temporels (« MOC »). Les données des collections ne sont pas ingérées dans le MocServer, uniquement les métadonnées. Et la liste des collections elle-même ne reprend pas la totalité des métadonnées disponibles, ni même l'exhaustivité des collections, notamment pour VizieR où les catalogues sans position ni information temporelle ne sont pas inclus. Pour utiliser le MocServer, nous vous recommandons de vous assurer que les collections et les propriétés sur lesquelles vous souhaitez travailler sont bien incluses. Si nécessaire, n'hésitez pas à utiliser à la place ou en complément les mécanismes de recherche générique proposés par les services originaux, et/ou un site d'implantation de l'annuaire IVOA.

Mémo : préciser qu'il y a également choix éditorial des métadonnées.

Propriétés des collections

Les propriétés associées à chaque collection de données sont représentées sous la forme d'un enregistrement classique de paires de « clé=valeur » regroupés dans un petit paragraphe ASCII. Elles reprennent la syntaxe utilisée pour les HiPS IVOA³.

Le MocServer n'utilise pas de liste de clés prédéfinies. Il prendra en compte toutes les clés des propriétés quelles qu'elles soient, même redondantes. Cependant, l'usage effectif des clés du MocServer s'appuie sur le modèle IVOA Obscore⁴ et en reprend une bonne part de son vocabulaire. Ainsi les propriétés couramment utilisées seront désignées par les mots clés suivants :

4 obs_label	= Label associated to the observations - ex: tabled
5 obs_title	= Observation title - ex: Herschel far-IR counterparts of SDSS galaxies (Dominguez+, 2014) (t..
6 obs_description	= Observation description - ex: Properties of SDSS-DR7/POP galaxies
7 obs_collection	= Collection of the observations - ex: GALEX survey
8 obs_collection_label	= Label of this collection - ex: GALEX

9 obs_release_date	= Observation release date - ex: 2015-05-04T12:04:45Z
10 dataproduct_type	= Type of data - ex: catalog

³ <https://www.ivoa.net/documents/HiPS/>

⁴ <https://www.ivoa.net/documents/ObsCore/>

11	bib_reference	= Bibliographic reference - ex: 2014MNRAS.441....2D
12	bib_reference_url	= URL to the bibliographic reference - ex: https://ui.adsabs.harvard.edu/?abs/2012A%26A...544A.156M
13	obs_ack	= Acknowledgement sentence - ex: Data products from observations made with ESO Telescopes at the La
14	obs_copyright	= Copy right mention - ex: IRAP/CADE
15	obs_copyright_url	= Url to copyright mention - ex: https://cdsarc.u-strasbg.fr/viz-bin/cat/J/MNRAS/441/2
16	t_min	= Start time of the observations (in MJD) - ex: 54965
17	t_max	= End time of the observations (in MJD) - ex: 56411
18	em_min	= Start wave length of the observations (in meters) - ex: 1E-5
19	em_max	= End wave length of the observations (in meters) - ex: 1E-4
20	obs_regime	= Regime - ex: Infrared

La liste exhaustive des mots clés gérés par le MocServer, le nombre de leurs occurrences, et un exemple de valeur associée pour chacun d'eux peut être obtenue en mentionnant le paramètre « [get=example](#) » en suffixe de l'URL du MocServer.

The screenshot shows a web browser window with the URL <https://alasky.cds.unistra.fr/MocServer/query?get=example>. The page displays a list of properties and their corresponding values for a specific observation. The properties include: obs_description, nb_rows, obs_regime, obs_astronomy_kw, vizier_popularity, data_jcd, cs_service_url, web_access_url, moc_access_url, bib_reference, obs_description_url, obs_copyright_url, dataproduit_type, obs_release_date, obs_label, moc_type, moc_sky_fraction, moc_order, obs_initial_ra, obs_initial_dec, and nhc_initial_fov. Each property is followed by its value and an example value in parentheses.

obs_description	= (3216x) ex: Properties of SUSS-UR//POP galaxies
nb_rows	= (28406x) ex: 123
obs_regime	= (26813x) ex: Infrared
obs_astronomy_kw	= (27278x) ex: Galaxies
vizier_popularity	= (28405x) ex: 0
data_jcd	= (28237x) ex: src.redshift
cs_service_url	= (30152x) ex: http://vizier.u-strasbg.fr/viz-bin/votable/-A?-source=J%2FMNRAS%2F4...
web_access_url	= (28405x) ex: http://vizier.u-strasbg.fr/viz-bin/VizieR-2?-source=J%2FMNRAS%2F441...
moc_access_url	= (28536x) ex: http://alasky.unistra.fr/footprints/tables/vizier/J_MNRAS_441_2_tab...
bib_reference	= (29731x) ex: 2014MNRAS.441....2D
obs_description_url	= (31099x) ex: https://cdsarc.u-strasbg.fr/viz-bin/cat/J/MNRAS/441/2
obs_copyright_url	= (29275x [+1:6]) ex: https://cdsarc.u-strasbg.fr/viz-bin/cat/J/MNRAS/441/2
dataproduit_type	= (29602x) ex: catalog
obs_release_date	= (28406x) ex: 2015-05-04T12:04:45Z
obs_label	= (28405x) ex: tabled
moc_type	= (28634x) ex: smoc
moc_sky_fraction	= (26132x) ex: 2.284E-6
moc_order	= (26119x) ex: 11
obs_initial_ra	= (26119x) ex: 163.35572842998585
obs_initial_dec	= (26119x) ex: 57.256701591949735
nhc_initial_fov	= (26119x) ex: 0.078679853431811713

<https://alasky.cds.unistra.fr/MocServer/query?get=example>

Identifications des collections

Chaque collection de données est désignée par un identificateur unique généré par le MocServer au moment de l'ingestion des données. La syntaxe de cet identificateur suit les recommandations IVOA décrites dans le document IVOA Identifier 1.12⁵ hormis le préfixe « *ivo://* ». Cet identificateur est créé en suivant la règle suivante⁶ :

1. Si un champ « [creator_did](#) » est mentionné dans les propriétés il sera pris tel que comme identificateur MocServer ;
2. A défaut, si un champ « [publisher_did](#) » est mentionné, il sera pris tel que comme identificateur ;
3. A défaut, le MocServer utilisera les champs « [publisher_id](#) » et « [obs_id](#) » et concaténera les deux valeurs pour générer un identificateur unique ;
4. A défaut, le MocServer utilisera le nom du fichier des « properties » où le caractère séparateur « / » est remplacé par un « _ », et l'extension du nom de fichier ignoré (ex : *CDS_P_DSScolor.prop* => *CDS/P/DSScolor*).

1	creator_did	= Full identifier of the observations (publisher+observation) - ex: ivo://ov-gso/P/HeIga/100
2	publisher_did	= Publisher identifier - ex: ivo://CDS
3	obs_id	= Observation identifier - ex: J/MNRAS/441/2/tabled

L'identificateur d'une collection est toujours ajouté aux propriétés de la collection sous la clé « [ID](#) » (en majuscules).

⁵ <https://www.ivoa.net/documents/latest/IDs.html>

⁶ Ces règles en cascade sont liées aux difficultés de convergence de l'IVOA. Elles couvrent l'ensemble des cas de figures.

Ex: `ID=CDS/P/DSScolor`

Interrogations basiques

Le MocServer supporte plusieurs modes d'interrogation dont le plus simple reprend les spécifications du standard IVOA « Simple Cone Search »⁷ et son extension temporelle⁸. Il reprend une partie de la syntaxe décrite dans ces documents.

Interrogation par cône

Une méthode basique pour interroger le MocServer consiste à lui décrire un cercle sur la sphère céleste au moyen des paramètres « `RA=ddd` », « `DEC=ddd` » et « `SR=ddd` ». RA et DEC expriment en degrés décimaux le centre du cercle dans le système de référence équatoriale ICRS, et SR indique en degrés le rayon du cercle.

Ex: `RA=10.8 DEC=32.2 SR=1.5`

Ces paramètres sont utilisés en suffixe de l'url du MocServer en prenant en compte les règles d'encodage classique http ('?' avant les paramètres, & en tant que séparateur, certains caractères éventuellement http encodés, notamment l'espace en « %2B »).

Ex: `https://alasky.cds.unistra.fr/MocServer/query?RA=10.8&DEC=32.2&SR=1.5`

Cette URL peut être utilisée directement dans un navigateur Web, ou bien par un outil tel que *wget* ou *curl*. Le résultat par défaut sera la liste des identificateurs des collections ayant au-moins une observation dans ce cône de recherche. La réponse à une telle interrogation ne prendra que quelques millisecondes quel que soit la taille du cercle de recherche⁹.

```
$ curl "https://alasky.cds.unistra.fr/MocServer/query?RA=10.8&DEC=32.2&SR=1.5"
CDS/B/assocdata/obscore
CDS/B/cb/lmxbdata
CDS/B/cfht/cfht
CDS/B/cfht/obscore
CDS/B/chandra/chandra
CDS/B/dao/obscore
CDS/B/eso/eso_arc
CDS/B/gcvs/gcvs_cat
CDS/B/gcvs/nsv_cat
CDS/B/gemini/obscore
```

Interrogation par polygone

En alternative au cône, le MocServer peut interpréter une interrogation par polygone sous la forme d'une expression STC-S, issue du standard « Simple Time Coordinate » de l'IVOA¹⁰. Le paramètre « `stc=xxx` » permet de désigner un polygone sur la sphère céleste au moyen d'une liste de couples « ascension droite », « déclinaison » exprimés en degrés décimaux dans le système de référence équatoriale ICRS.

Ex: `stc=Polygon 57.376 24.053 56.391 24.622 56.025 24.049 56.616 24.290`

⁷ <https://www.ivoa.net/documents/cover/ConeSearch-20080222.html>

⁸ <https://www.ivoa.net/documents/ConeSearch/20200828/index.html>

⁹ Dans la suite du document, par souci de concision, le préfixe de l'URL sera omis (« http ... ? »), seuls les paramètres seront mentionnés.

¹⁰ <http://www.ivoa.net/Documents/latest/STC-S.html>

A noter que le MocServer supporte uniquement les expressions STC-S basiques spatiales à savoir le *Polygon* et le *Circle*, sans prendre en compte des opérations d'union ou d'intersection.

Interrogation par intervalle temporel

Lorsque la requête porte, non pas sur une région du ciel, mais sur un intervalle de temps, le paramètre à utiliser sera « **TIME=mjd1 mjd2** ». *mjd1* représente la date de début, et *mjd2* la date de fin, toutes deux exprimées en « Modified Julian Date ». Pour convertir une date classique en format MJD, vous pouvez vous aider de nombreux convertisseurs disponibles sur le Web¹¹.

Ex : **TIME=39619.44097 59977.7875**

Ainsi, la requête ci-dessus va retourner toutes les collections dont au-moins une observation a eu lieu entre le 9 mai 1967 à 10:35 et le 2 février 2023 à 18:54.

```
$ curl "https://alasky.cds.unistra.fr/MocServer/query?TIME=39619.44097+59977.7875"
CDS/B/astorb/astorb
CDS/B/comets/comets
CDS/C/CALIFA/V500/DR2
CDS/C/CGPS-HI
CDS/C/GALFAHI/Narrow
CDS/C/GALFAHI/Narrow/DR2
CDS/C/GALFAHI/wide/DR2
```

Contraintes spatiales et temporelles

Le critère de recherche spatial et celui temporel peuvent être couplés. Seules les collections répondant simultanément aux deux critères seront retenues.

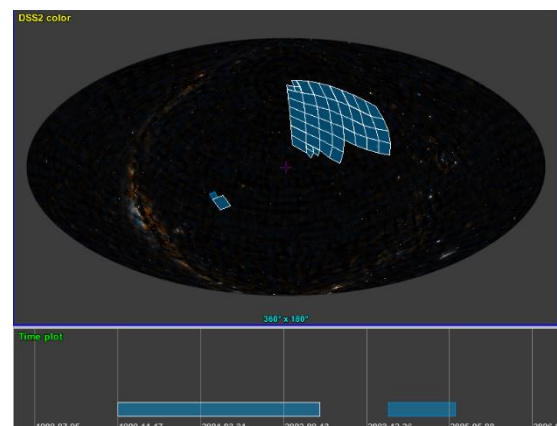
Le paramètre « **intersect=xxx** » permet de préciser si la zone spatio-temporelle de la requête doit : « **enclosed** » - être complètement incluse dans celle des collections à retenir, « **covers** » - la recouvrir totalement, ou « **overlaps** » - simplement avec une intersection non nulle (le mode par défaut).

Lorsque le MocServer ne connaît pas la couverture spatiale (resp. temporelle) d'une collection, celle-ci n'est pas retenue.

Interrogation par MOC

Alternativement aux interrogations par cônes, polygones et intervalles temporelles, le MocServer supporte également l'interrogation directe par MOC. Par cette méthode il est possible de décrire très précisément le critère spatio-temporel de l'interrogation. La description de ce MOC d'interrogation pourra être faite « in line », ou via une « url » ou même « uploadé ».

MOC inline : Le paramètre « **moc=xxx** » permet de décrire un MOC ASCII suivant le standard IVOA MOC 2¹².



¹¹ Convertisseur NASA-HEASARC : <https://heasarc.gsfc.nasa.gov/cgi-bin/Tools/xTime/xTime.pl>

¹² <https://ivoa.net/documents/MOC/>

Ainsi l'exemple ci-dessous va interroger le MocServer dans deux régions spatiales disjointes chacune d'elle associée à un intervalle temporel spécifique (cf. ci-contre).

Ex: `moc= t16/6020-6022 s3/128 4/345-510 t16/6024 s4/514-516`

MOC via URL : Pour utiliser des MOC sérialiser dans un fichier, le paramètre « `url=xxx` » permet d'indiquer l'adresse Web du MOC qui sera utilisé pour l'interrogation du MocServer. Le MocServer supporte la représentation MOC ASCII, FITS et même JSON (cf. document IVOA MOC 2.0 et son annexe).

Ex: `url=http://alasky.u-strasbg.fr/SDSS/DR9/color/Moc.fits`

MOC « uploadé » : Le MocServer peut également prendre en compte un MOC téléchargé (« uploadé ») via une requête POST multipart « `moc` » répondant au paramètres HTML suivants :

```
<FORM enctype="multipart/form-data" method="post">
  <input name="moc" type="file"/>
  ...
</FORM>
```

5) File MOC search

MOC server can be queried by a local MOC file. The search coverage is a local MOC uploaded via Multipart POST method (multipartID: moc).
MOC must be a local file (FITS, ASCII or JSON syntax)

Aucun fichier sélectionné.

La génération et la manipulation des MOC spatiaux, temporels, ou spatio-temporels s'appuient sur plusieurs librairies disponibles en python¹³, java¹⁴, ... ainsi que sur des logiciels, notamment Aladin Desktop¹⁵.

Référentiels spatiaux alternatifs : Les MOC célestes sont toujours exprimés dans le référentiel équatorial ICRS. Cependant, le MocServer gère également des collections associées aux planètes, et pour chacune d'elle, le MOC s'exprime dans le référentiel de la planète. Le paramètre « `spacesys=xxx` » permet d'indiquer le référentiel spatial (*equatorial, mercury, earth, moon*¹⁶, etc) du MOC passé en paramètre. Les collections ne relevant pas du même référentiel spatial ne seront pas pris en compte¹⁷.

¹³ <https://github.com/cds-astro/mocpy>

¹⁴ <https://wiki.ivoa.net/twiki/bin/view/IVOA/MocInfo>

¹⁵ <https://aladin.cds.unistra.fr/AladinDesktop/>

¹⁶ La syntaxe pour décrire les référentiels spatiaux des planètes n'est pas encore définie par l'IVOA. Le MocServer utilise actuellement le nom de la planète/corps en langue anglaise et en minuscules.

¹⁷ A noter que le paramètre « `timesys=xxx` » est déjà implanté. Il permettra de spécifier le référentiel temporel du MOC passé en paramètre. A l'heure actuelle, le MocServer ne contient que des données dont les informations temporelles sont exprimées dans le système temporel TCB (Barycentric Coordinate Time).

Interrogation par propriétés

L'interrogation du MocServer peut également prendre en compte des critères classiques de sélection portant sur les propriétés associées à chaque collection de données. Ces filtrages peuvent être spécifiés avec ou sans contraintes spatiales et/ou temporelles associées.

A noter que les valeurs indiquées dans les filtres ci-dessous font la différence entre majuscules et minuscules. Il est possible de s'en affranchir au moyen du paramètre « `casesensitive=false` ».

Méthode basique par adjonction de paramètres à l'URL

L'ajout d'une contrainte sur une propriété peut s'effectuer en ajoutant directement à l'URL de requête la clé du champ et la contrainte sur ce champ sous la forme d'un paramètre HTTP.

L'utilisation des jokers est possible : '*' – n'importe quelle suite de caractères, même vide, '?' – n'importe quel caractère. L'exemple ci-dessous sélectionne toutes les collections ayant une propriété « `obs_title` » dont la valeur contient le mot « *NASA* ».

Ex : `obs_title=*NASA*`

Cette contrainte sera accolée à l'URL de requêtes du MocServer. Si d'autres contraintes spatiales ou temporelles sont indiquées, les collections sélectionnées devront remplir toutes les conditions.

Ex : `https://.../query?RA=10.8&DEC=32.2&SR=1.5&obs_title=*NASA*`

Jokers sur les clés : Les jokers '*' et '?' peuvent également être utilisés pour désigner un ensemble de clés. Ainsi l'exemple ci-dessous recherchera le mot « *ESO* » dans l'ensemble des champs commençant par « `obs_` ».

Ex : `obs_*=*ESO*`

Valeurs multiples : L'utilisation du séparateur ',' (virgule) permet d'indiquer plusieurs alternatives de filtrage pour un même champ. L'exemple ci-dessous recherchera dans le champ « `obs_regime` » la valeur « *Infrared* » ou « *Radio* ».

Ex : `obs_regime=Infrared,Radio`

Clés multiples : De la même façon que pour les valeurs multiples, le séparateur ',' (virgule) permet d'indiquer plusieurs clés concernées par le filtre, avec ou sans recours aux jokers ('*' et '?'). Il suffit que l'un des champs concernés remplisse la condition pour que la collection soit retenue (logique OU implicite).

Ex : `obs_title,obs_collection=*ESAC*`

Contraintes multiples : Lorsque plusieurs contraintes sont à valider simultanément, il sera nécessaire d'indiquer autant de paramètres « `clé=valeur` » que de contraintes (logique ET implicite). L'exemple ci-dessous, contrairement à l'exemple précédent, retiendra les collections dont le mot « *ESAC* » est présent à la fois dans le champ « `obs_title` » et dans le champ « `obs_collection` ».

Ex : `obs_title=*ESAC* obs_collection=*ESAC*`

Complémentaire : Le caractère '!' (point d'exclamation), en préfixe de la valeur, indiquera le résultat complémentaire. Ceci revient à sélectionner les collections dont le champ indiqué **ne contient pas** la chaîne recherchée. **A noter que le caractère '!' est bien placé en préfixe de la valeur**, et non avant le signe '='.

Ex : `obs_id=!*CDS*`

Astuce : La sélection des collections dont les propriétés **ne** contiennent **pas** un champ particulier sera donc possible en utilisant le filtrage « `cle=!*` ».

L'utilisation du complémentaire sur un filtrage spécifiant des clés-multiples (par liste ou par jokers) est déconseillée car facilement sujet à mauvaise interprétation de la logique booléenne appliquée. Il est préférable d'utiliser la méthode par « expression algébrique explicite » décrite dans la section suivante.

Comparaisons numériques : Dans le cas où les valeurs sont des nombres, il est possible d'utiliser les fonctions de comparaison : '`>`' - supérieur, '`<`' - inférieur, '`>=`' - supérieur ou égal et '`<=`' - inférieur ou égal, ou encore un intervalle de valeurs via le séparateur « `..` » (deux points). Dans le cas des fonctions de comparaison elle sera indiquée en préfixe de la valeur, et non en remplacement du caractère '=' (spécifique à la syntaxe des paramètres des URLs). Les éventuelles valeurs non numériques ne seront jamais retenues.

Ex 1 : `em_min=>10`

Ex 2 : `t_min=52000..54000`

Comparaisons calendaires : De la même manière que pour les nombres, les champs utilisés pour des dates peuvent également être comparés. Les formats de dates reconnus sont ISO, avec ou sans indication des heures et minutes (`yyyy-mm-dd` et `yyyy-mm-ddThh:mm:ss`), les dates usuelles (`dd/mm/yyyy`), ou encore l'année (`yyyy`). Les valeurs non calendaires ne seront pas prises en compte.

Désambiguïsé : Dans le cas où le filtre sur une propriété utilise un mot clé qui serait également un des paramètres du MocServer, à savoir *RA*, *DEC*, *SR*, *TIME*, *stc*, *moc*, *url*, *spacesys*, *timesys*, *fmt*, *get*, *fields*, *casesensitive*, *intersect*, *MAXREC*, *expr*, *op*¹⁸, le mot clé devra être précédé du préfixe « *filter.* »

Ex1 : `filter.fmt=*jpeg*`

Ex2 : `filter.TIME=13:* & TIME=3969 59977`

Méthode avancée par expression algébrique

Lorsque les règles de filtrages deviennent plus complexes, il est préférable d'opter pour la méthode avancée qu'autorise le paramètre « `expr=xxx` ». Il permet de spécifier une expression algébrique complète ayant recours aux opérateurs ensemblistes classiques : « `||` » - union, « `&&` » - intersection, « `!&` » - soustraction, et aux mécanismes de parenthèses. Tout comme pour la méthode basique, l'utilisation de jokers et de listes est supportée.

L'exemple ci-dessous retiendra les collections dont l'identificateur (ID) commence par « *xcatdb* » ou pour lesquelles le régime (*obs_regime*) est soit « *X-ray* », soit « *UV* », disposant d'une URL HiPS (*hips_service_url*) à l'exception des collections dont l'un des champs commençant par « *obs_* » contient le mot « *CONSTEL* ».

`expr=((ID=xcatdb* || obs_regime=X-ray,UV) && hips_service_url=*) &! obs_*=*CONSTEL*`

¹⁸ Ainsi que *SIZE*, *POS* et *callback* toujours supportés pour compatibilité avec les versions précédentes du MocServer.

Résultat de la requête

Par défaut, le MocServer retourne les collections concernées par des contraintes spatiales, temporelles et/ou sur les propriétés, toujours triées par ordre lexicographique croissant des ID des enregistrements. Le paramètre « [get=xxx](#) » permet de préciser la forme de ce résultat. Ce peut être une liste d'identificateurs, le nombre de collections, ou les propriétés des collections. Ces résultats sont fournis par défaut au format ASCII. L'alternative JSON est également possible (cf. fin de ce paragraphe).

Liste des identificateurs. Les identificateurs des collections sélectionnées sont retournés sous la forme d'une liste, un identificateur par ligne (cf. paragraphe ci-dessus). C'est le mode par défaut, équivalent à l'utilisation du paramètre « [get=id](#) »

Nombre de collections. L'utilisation du paramètre « [get=number](#) » va restreindre le résultat au nombre de collections concernées.

Ex: RA=10.8 DEC=32.2 SR=1.5 [get=number](#)

```
$ curl "https://alasky.cds.unistra.fr/MocServer/query?RA=10.8&DEC=32.2&SR=1.5&get=number"
1889:
```

Propriétés des collections. L'utilisation du paramètre « [get=record](#) » retourne, pour chaque collection concernée, la liste de ses propriétés. Le résultat se présente sous la forme d'enregistrements de « *clé=valeur* » comme décrit ci-dessus. Chaque enregistrement est séparé du précédent par une ligne vide. L'utilisation du paramètre « [fields=xxx](#) » permet de spécifier explicitement les champs souhaités sous la forme d'une liste de clés séparées par des virgules. L'utilisation de caractères « jokers » est également possible : « *** » – n'importe quelle suite de caractères (même vide), « *?* » – n'importe quel caractère.

Ex: RA=10.8 DEC=32.2 SR=1.5 [get=record fields=ID,obs_tit*](#)

```
$ curl "https://alasky.cds.unistra.fr/MocServer/query?RA=10.8&DEC=32.2&SR=1.5&get=record&fields=ID,obs_tit*"
ID
obs_title      = CDS/B/assocdata/obscure
                = Associated data in VizieR (G.Landais, 2016) (obscure)
ID
obs_title      = CDS/B/cb/lmxbdata
                = Cataclysmic Binaries, LMXBs, and related objects (Ritter+, 2004) (lmxbdata)
ID
obs_title      = CDS/B/cfht/cfht
                = Log of CFHT Exposures (CADC, 1979-) (cfht)
ID
obs_title      = CDS/B/cfht/obscure
                = Log of CFHT Exposures (CADC, 1979-) (obscure)
```

Limitation du nombre de réponses. Le paramètre « [MAXREC=nn](#) », suivant le standard IVOA « Simple Cone Search » permet de limiter le nombre de réponses attendues.

Formats alternatifs : Comme décrit ci-dessus, le format d'une réponse est par défaut de l'ASCII. Le paramètre « [fmt=json](#) » permet d'obtenir le même résultat mais empaqueté en JSON.

Ex: ... [get=record fields=ID,obs_tit* fmt=json](#)

```
$ curl "https://alasky.cds.unistra.fr/MocServer/query?RA=10.8&DEC=32.2&SR=1.5&get=record&fields=ID,obs_tit*&fmt=json"
[
  { "ID": "CDS/B/assocdata/obscure", "obs_title": "Associated data in VizieR (G.Landais, 2016) (obscure)" },
  { "ID": "CDS/B/cb/lmxbdata", "obs_title": "Cataclysmic Binaries, LMXBs, and related objects (Ritter+, 2004) (lmxbdata)" },
  { "ID": "CDS/B/cfht/cfht", "obs_title": "Log of CFHT Exposures (CADC, 1979-) (cfht)" },
  { "ID": "CDS/B/cfht/obscure", "obs_title": "Log of CFHT Exposures (CADC, 1979-) (obscure)" },
  { "ID": "CDS/B/chandra/chandra", "obs_title": "The Chandra Archive Log (CXC, 1999-2014) (chandra)" },
  { "ID": "CDS/B/dao/obscure", "obs_title": "DAO Science Archive observations (CADC, 2020) (obscure)" },
  { "ID": "CDS/B/eso/eso_arc", "obs_title": "ESO Science Archive Catalog (ESO, 1991-2022) (eso_arc)" },
  { "ID": "CDS/B/gcvs/gcvs_cat", "obs_title": "General Catalogue of Variable Stars (Samus, 2007-2017) (gcvs_cat)" }
]
```

Attention : Dans le cas de champs redondants (ex : *obs_regime=X-ray, obs_regime=UV*), la valeur sera représentée sous forme d'une liste suivant la syntaxe JSON (ex : `["X-ray", "UV"]`)

Résultats sous forme de couverture MOC

En alternative aux identificateurs ou enregistrements des collections sélectionnées, le paramètre « `get` » permet de demander au MocServer de retourner la couverture des collections retenues sous la forme d'un MOC. L'exemple le plus évident consiste à demander la couverture d'une collection spécifique désignée par son identificateur, ce qui ramène à utiliser le MocServer comme un « serveur de Moc basique ».

Ex : ID= CDS/P/SDSS9/u `get=moc`

Règles de génération : La construction du MOC généré en tant que résultat d'une requête sur le MocServer suit par défaut les règles ci-dessous :

1. Le MOC retourné sera l'union de l'ensemble des couvertures des collections retenues
2. Si d'autre part, des contraintes spatiales et/ou temporelles ont été spécifiées, le résultat final sera l'intersection du MOC calculé à l'étape 1 avec celui du MOC issue des contraintes spatiales et/ou temporelles.

Le paramètre « `op=xxx` » permet de modifier ce comportement par défaut en indiquant explicitement « `union` » - l'union des MOC concernés, ou « `intersection` » - l'intersection.

Spécialisation du MOC : Le paramètre « `get=xxx` » permet de préciser quel type de MOC doit être retourné, MOC spatial (SMOC), temporel (TMOC) ou spatio-temporel (STMOC). Ce paramètre pourra prendre les valeurs suivantes : « `anymoc` » - un MOC dont la nature dépend des collections sélectionnées, « `smoc` » - uniquement la composante spatiale du MOC résultat, « `tmoc` » - uniquement la composante temporelle du MOC résultat, « `stmoc` » - un MOC spatio-temporel quel que soit la nature du MOC résultat. Dans ce dernier cas, si l'un des MOCs qui constituent la réponse n'a pas de dimension spatiale (resp. temporelle), celle-ci sera considérée comme totale, à savoir tout le ciel (resp. à n'importe quelle date).

Pour compatibilité avec les versions précédentes du MocServer, la valeur « `moc` » est un synonyme de « `smoc` » et retournera uniquement la composante spatiale du MOC résultat.

Formats alternatifs : Le format du MOC est par défaut du « FITS » - recommandé par le document MOC 2.0 IVOA. Celui-ci peut être modifié par le paramètre « `fmt=xxx` ». Il peut prendre les valeurs : `ascii` – format MOC ASCII ou `json` - format MOC JSON.

Résolution : Le MOC retourné utilise par défaut la résolution des MOCs stockés dans le MocServer. Elle est déterminée par son « *MOC order* » dans la terminologie MOC. Il est possible de modifier cette résolution au moyen du paramètre « `order=xxx` » où `xxx` indique le ou les « MOC orders ». Ceux-ci peuvent être spécifiés explicitement par une valeur et/ou indirectement en indiquant une taille maximale du MOC résultant. L'utilisation du préfixe '`s`', respectivement '`t`', permettra de différencier l'ordre MOC spatial, et l'ordre MOC temporel. Pour compatibilité avec les versions précédentes du MocServer, sans préfixe, il s'agira de l'ordre MOC spatial. La taille maximale est indiquée par le caractère '`<`' suivi d'une valeur et d'une unité (sans espace entre les éléments). Les « orders » seront automatiquement dégradés jusqu'à ce que le MOC retourné ait un volume mémoire inférieur à la limite indiquée (unités reconnues : `MB`-megabyte, `KB`-kilobyte).

Ex: ID= CDS/P/SDSS9/u get=stmoc fmt=ascii order=s3 t45 <10KB

Synchronisation des données pour un annuaire tiers

Le MocServer peut être également utilisé comme source d'un annuaire tiers. Il s'agira pour une application tierce de récupérer la liste des enregistrements du MocServer, et de disposer d'un moyen efficace pour la maintenir à jour (cf. section suivante).

L'utilisation de la requête POST multipart « [maj](#) » permettra d'uploader un fichier ASCII contenant l'ensemble des ID des collections précédemment téléchargées.

```
<form enctype="multipart/form-data" method="post">
  <input name="maj" type="file">
</form>
```

Le MocServer retournera l'ensemble des enregistrements des collections non encore recensées, sous la forme de paragraphes « *clé=valeur\nclé=valeur\n...* », séparés par une ligne vide. Il fournira également la liste des enregistrements qui ont été supprimés en leur associant le champ « [MOCSERVER_REMOVE=true](#) ».

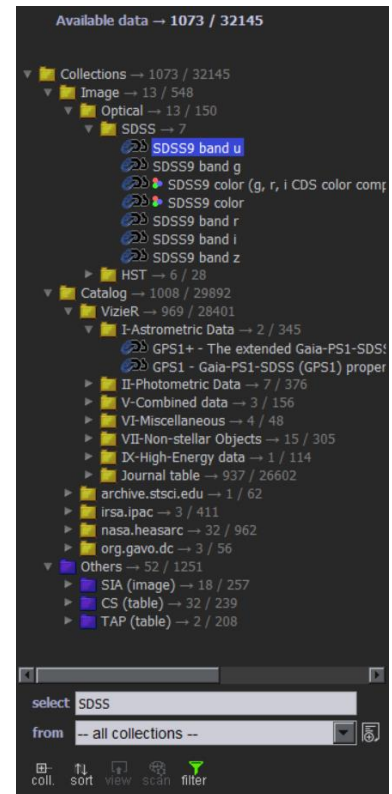
L'adjonction d'un estampillage à chacun des ID soumis (ex : *CDS/P/DSScouleur=1664225262000*) ajoutera à la liste des nouveaux enregistrements retournés, ceux dont la date est postérieure à celle indiquée. A noter que le MocServer ajoute systématiquement aux propriétés des collections un champ « [TIMESTAMP](#) » fournissant la date de mise à jour de chaque enregistrement (en millisecondes depuis le 1^{er} janvier 1970). C'est cet estampillage que l'application tierce devra réutiliser pour s'assurer de la mise à jour des enregistrements.

Enfin, si le fichier uploadé se réduit à une unique date d'estampillage, ce seront tous les enregistrements postérieurs qui seront retournés.

Utilisation du MocServer par Aladin Desktop

Le logiciel Aladin Desktop est très fortement lié au MocServer et la plupart des fonctionnalités du MocServer sont accessibles via l'interface graphique d'Aladin.

Ainsi à chaque démarrage, Aladin dialogue avec le MocServer pour récupérer l'ensemble des propriétés des collections et en construit un arbre graphique permettant à l'utilisateur d'explorer les données disponibles. Lorsque l'utilisateur d'Aladin visualise une région du ciel, cet arbre est mis en couleur pour indiquer en vert les collections ayant au-moins une observation dans le champ en cours de visualisation, et en orange celles qui n'en ont pas. Les branches restant blanches révèlent les collections pour lesquelles le MocServer ne dispose pas de MOC spatiaux associés. De la même manière, si l'utilisateur visualise un graphique temporel, l'arbre des données sera colorié en fonction de l'intervalle de temps présenté.



Exemples d'utilisation

La liste des exemples ci-dessous sont repris de l'interface du MocServer, et accessible via l'url suivante :

<https://alasky.cds.unistra.fr/MocServer/example>

- **Around M31** - IDs of all data sets in 5 deg around M31

Returns the ID list of all data sets overlapping a circle of 10 degree diameter centered on M31

→ <http://.../MocServer/query?RA=10.67305&DEC=41.26875&SR=5>

- **In 2000** - IDs of all data sets in 2000

Returns the ID list of all data sets having on observation during the year 2000

→ <http://.../MocServer/query?TIME=51544+51910>

- **Overlapping a STC polygon** - IDs of all data sets overlapping a polygon

→ <http://.../MocServer/query?stc=Polygon+57.376+24.053+56.391+24.622+56.025+24.049+56.616+24.290>

- **Containing a STC polygon** - IDs of all data sets fully containing a polygon

Returns the ID list of all data sets which contains the area described by the STC polygon

→

<http://.../MocServer/query?stc=Polygon+312.39906+31.76699++312.66095+31.53000+312.64826+31.01805+312.45437+30.43480+312.32075+30.75116+312.28911+31.11123+312.21508+31.50335&intersect=enclosed>

- **HST data sets over SDSS** - IDs of HST collections overlapping SDSS observations

Returns the ID list of all data sets overlapping SDSS MOC (downloading by URL) and provides those for which their ID contains the substring "P/HST/"

→ http://.../MocServer/query?ID=*P/HST/*&url=http%3A%2F%2Falasky.unistra.fr%2FSDSS%2FDR9%2Fcolor%2FMoc.fits

- **Overlapping an STMOC** - IDs of all data sets overlapping an inline STMOC

Returns the ID list of all data sets overlapping the inline MOC t16/6020-6022 s3/128 4/345-510

→ <http://.../MocServer/query?moc=t16%2F6020-6022+s3%2F128+4%2F345-510>

- **SCUBA2 spatial coverage** - Spatial MOC of SCUBA2 observations

Provides the spatial MOC of the SCUBA2 850em observations. The MOC is provided in FITS format. For displaying the result, clic and drag this URL in Aladin Desktop.

→ <http://.../MocServer/query?ID=CDS/P/SCUBA2/850em&get=smoc>

- **GALEX & HERSCHEL spatial union** - Spatial MOC of the GALEX and HERSCHEL coverage

Provides the union of spatial MOCs of GALEX-NUV and HERSCHEL-PACS observations

→ <http://.../MocServer/query?ID=CDS/P/GALEXGR6/AIS/NUV,ESA/VO/P/HERSCHEL/PACS-color&get=smoc>

- **A&AS spatial coverage** - Spatial MOC of all A&AS tables

Provides the union of spatial MOCs of all tables published in A&AS. The resulting MOC is provided in JSON format (not IVOA standard)

→ http://.../MocServer/query?ID=CDS/J/A%2BAS/*&get=smoc&fmt=json

- **AJ temporal coverage** - Temporal MOC of all Aj tables

Provides the union of temporal MOCs all all tables published in AJ.

→ http://.../MocServer/query?ID=CDS/J/AJ*&get=tmoc

- **SDSS & GALEX spatial intersection** - Spatial MOC of the intersection of SDSS coverage and GALEX coverage

Provides the spatio-temporal MOC of the LAMOST DR4 observations. This STMOC is provided in FITS format

→ http://.../MocServer/query?ID=*SDSS9/g,*GR6/AIS/FUV&get=smoc&op=intersection

Provides the spatio-temporal MOC of all coincidental observations in time and space of LAMOST & ALMA calibrator cat observations

- **LAMOST & ALMA calibrator cat co-coincidental coverage** - Spatio-temporal MOC of the LAMOST & ALMA calibrator cat coincidental observations

Provides some properties (ID, description, sky fraction) of all tables covering an half sky or more. The output

format is JSON.

→ http://.../MocServer/query?moc_sky_fraction=>0.5&ID=!*/P/*&fields=obs_id,obs_title,moc_sky_fraction&fmt=json

- **Copyright filtering** - Advanced record filtering based on field names with wildcards

Provides the records for which one copyright fields contains the word NASA or ESA, and a field contains the word DSS

→ http://.../MocServer/query?*copyright*=*NASA*,*ESA*&*=*DSS*&get=record

- **URL filtering** - HiPS URLs search

Provides the records of HiPS distributed simultaneously by saada and alasky http server

→ http://.../MocServer/query?hips_service_url*=http://saada.*&hips_service_url*=http://alasky.*&get=record

- **HiPS colored surveys** - HiPS colored surveys access

Provides the list of available HiPS colored surveys (ID & base URL) in JSON format

→ http://.../MocServer/query?hips_service_url=*&datapoint_subtype=color&fields=id,hips_service_url&fmt=json

- **HiPS DSS colored** - DSS colored HiPS mirror sites

Provides the list of sites publishing the HiPS DSS colored

→ http://.../MocServer/query?ID=CDS/P/DSS2/color&fields=hips_service_url

- **HiPS Gaia resources** - Full set algebraic language filtering

Provides the ID list of HiPS Gaia data sets except simulations thanks to this expression: ((obs_*=*gaia* || ID=*gaia*) && hips_service_url=*) &! obs_*=*simu*

→

http://.../MocServer/query?expr=%28%28obs_*%3D*gaia*+%7C%7C+ID%3D*gaia*%29+%26%26+hips_service_url%3D*%29+%26%26+obs_*%3D*simu*%26casesensitive%3Dfalse

Administration du MocServer

Cette section est destinée aux personnes en charge de l'administration d'un MocServer ou intéressées par la technologie utilisée pour implanter un tel service.

Technologie et déploiement

Le Mocserver utilise une classique technologie « servlet java » exploitée par un serveur Tomcat. Le fonctionnement de cette servlet requière uniquement la présence d'un répertoire dédié qui contient l'ensemble des données nécessaires à son fonctionnement, principalement les fichiers [properties](#) et des fichiers [MOCs](#) qu'il doit gérer, ainsi que le ou les [scripts](#) nécessaires à la mise à jour de ces fichiers.

Configuration. Un petit fichier de configuration optionnel « [mocserver.conf](#) » permet de spécifier le mode de fonctionnement du MocServer et notamment s'il s'agit d'une implantation « [master](#) » - met à jour périodiquement les fichiers [properties/MOC](#) au moyen des scripts sus-cités, ou « [slave](#) » - se contente d'utiliser les données tels que. Dans le cas d'un serveur « master », la périodicité des mises à jour, le nombre de backup est également configurable ainsi que la taille maximale autorisée en mémoire pour chaque MOC (cf. ci-contre).

```
alasky:/etc$ more mocserver.conf
# mocserver.conf => must be stored in /etc (and if not in /data/Moc)

# Harvest mode (manual|auto|slave) => default manual
# manual: harvesting process is controlled manually via Admin Web form
# auto: harvesting process is automatic and can be controlled via Admin Web form
# slave: MocServer is running as slave => use/reuse /data/Moc/multimoc.bin as it is. No Admin Web control form
HarvestMode = auto

# In case of harvesting (no slave mode)...
# Delay between two automatic harvests (in sec - default 86400) => see HarvestMode = auto
#HarvestDelay = 259200
# Maximum MB per MOC in RAM (=> decrease the MocOrder if too big - default 30)
MaxMemPerMoc = 10
# Number of Multimoc.bin backups (default 9)
#MaxBackup = 9
# Number of Pilot.html history (default 9)
MaxHistory = 20
```

Mise-à-jour. Le renouvellement du contenu du MocServer est effectué au moyen d'un unique script « [harvest.bat](#) » qui doit assurer la mise-à-jour des fichiers de [properties](#) et des [MOCs](#) à gérer. Par défaut, celui-ci est exécutée une fois toutes les 24h. Il est supposé créer ou recopier l'ensemble des fichiers concernés dans le répertoire courant qui s'avère être un répertoire temporaire « [Pilot](#) ». Les noms de ces fichiers doivent impérativement être construits à partir de l'ID des collections concernées, en remplaçant le caractère « / » par un « _ », et en utilisant l'extension « [.prop](#) » pour les [properties](#) et « [.fits](#) » pour les [MOCs](#)¹⁹.

```
-rw-r--r-- 1 tomcat7 tomcat7 48960 Oct 12 2021 CDS_B_chandra_chandra.fits
-rw-r--r-- 1 tomcat7 tomcat7 1125 Mar 9 05:22 CDS_B_chandra_chandra.prop
```

A la fin de l'exécution du script, le MocServer va comparer les données dont il dispose en mémoire et celles fournies par le répertoire temporaire. Puis, si la comparaison est validée, c'est-à-dire ne relève pas de trop nombreuses suppressions de données²⁰, le MocServer va substituer le répertoire des données actives « [Data](#) » avec ce répertoire temporaire « [Pilot](#) » et recharger son contenu mémoire à partir de ce nouveau jeu de données. A contrario, le MocServer ne rendra pas effective la mise-à-jour, et le signalera sur l'interface d'administration décrite ci-dessous pour une vérification et un acquiescement manuel.

Mode automatique ou manuel. L'opération de mise-à-jour automatique s'effectuant sans interruption de service elle requière le double de mémoire RAM. C'est pourquoi il est indispensable de bien

¹⁹ A noter que la syntaxe du MOC elle-même peut être indifféremment FITS, ASCII ou JSON.

²⁰ Le MocServer estime « suspecte » une mise-à-jour supprimant plus de 10 collections issues d'une même autorité.

Données permanentes. En complément au répertoire « [Data](#) », le répertoire « [Permanent](#) » peut être utilisé pour y mémoriser les [properties](#) et [MOCs](#) de collections dites « permanentes ». Elles seront systématiquement ajoutées aux collections issues du répertoire « [Data](#) » et le cas échéant remplaceront les informations des collections ayant le même identificateur.

```
client_category = Image/Gamma*
>obs_regime = Gamma-ray

ID = ESAVO*
>client application = -AladinDesktop
```

Backups et logs. Lors d'une mise-à-jour, le MocServer conserve le contenu mémoire précédent sous la forme d'un unique fichier binaire « [Multimoc.bin](#) » généré dans le répertoire « [Cache](#) ». Ce fichier de « backup » ainsi que les 9 précédentes versions pourront être réutilisés via l'interface d'administration pour revenir immédiatement à un contenu antérieur. Le MocServer conserve également dans ce même répertoire le « log » du script ainsi que le résultat de la comparaison effectuée pour valider la mise-à-jour sous les noms respectifs « [HarvestLog.html](#) » et « [Pilot.html](#) ». Ceux-ci-ci, ainsi que les 20 versions antérieures sont consultables par l'interface d'administration.

Pilot comparison - 2023-03-03 23:18:56...

Current multiMoc: 32637 datasets using 3GB (49MB for properties) [26195 SMoc + 2521 TMoc + 1166 STMoc]
Pilot: 32647 datasets using 49MB [0 SMoc + 0 TMoc + 0 STMoc]

0 deleted, 10 added, 1347 modified, 31290 unchanged

- + CDS/3/ApJ/919/85/table1
- + CDS/3/ApJ/919/85/table14
- + CDS/3/ApJ/919/62/table4
- + CDS/3/ApJ/919/62/table3
- + CDS/3/ApJ/919/62/table5
- + nasa.heasarc.m83xrbco
- + CDS/3/ApJ/919/99/table3
- + CDS/3/ApJ/919/62/sample
- + CDS/3/ApJ/919/85/calibr1
- + CDS/3/AJ/164/244/table2
- + 1339 records with "vizier_popularity" modified (see below)
- + nasa.heasarc.m83cxo (get it)
 - * obs_description = The authors of this table have obtained a series of deep X-ray images of the nearby (4. =====> The authors of this table have obtained a series of deep X-ray images of the nearby (4.
- + CDS/Simbad (get it)
 - * hips_release_date = 2023-03-02T03:06Z ==> 2023-03-03T03:07Z
- + UNK_AUTH/P/1673397445, APPEND/P/1673576470, APPEND/P/1673576470, APPEND/P/1677782867 (get it)
 - datapoint_subtype
 - + hipsgen_date_7
 - + hipsgen_params_7
 - * hips_estsize = 7312620 ==> 12883575
 - * addendum_id = APPEND/P/1673397445, APPEND/P/1673576470
 - =====> APPEND/P/1673397445, APPEND/P/1673576470, APPEND/P/1677782867

Interface d'administration

Le MocServer dispose d'une interface d'administration distante, à accès restreint, accessible par l'url suivante :

<https://alasky.cds.unistra.fr/MocServer/admin>

Cette interface d'administration du MocServer fournit une brève description de la localisation des répertoires et fichiers de configuration décrits précédemment, et dont un exemple d'implantation est donné ci-dessous :

Configuration

- **Configuration file:** /etc/mocserver.conf ([optional, in /etc otherwise in /data/Moc] allowing to modify default MocServer behavior)
- **Data directory:** /data/Moc/Data (directory containing current MOCs & Props used by the MocServer as a list of individual files)
- **Pilot directory:** /data/Moc/Pilot (directory dedicated to MOCs & Props in preparation for the next release)
- **Permanent directory:** /data/Moc/Permanent (directory containing local MOCs & Props, not harvested - overriding possible duplicated records)
- **Harvest script:** /data/Moc/harvest.bat (script called for copying/downloading MOCs & Props in Pilot directory)
- **Rewriting rules:** /data/Moc/exceptions.prop (rewriting rule exceptions for modifying some prop records during MocServer (re)loading process)
- **Cache directory:** /data/Moc/Cache (directory containing binary files (MOC&Props in one file) & statistics)

L'interface fournit également les statuts courants du serveur Tomcat et de la servlet MocServer (cf. exemple présenté ci-dessous), ainsi que les statistiques d'utilisation hebdomadaire :

32705 datasets using 3GB (50MB for properties) [26328 SMoc + 2566 TMoc + 1166 STMoc]

Servlet status

- Apache Tomcat/7.0.88 (Debian)
- Running servlets: 4
- Number of threads: 106
- Memory size: 9.46GB
- Free memory: 3.51GB

Enfin, cette interface permet d'effectuer manuellement les différentes étapes d'une mise-à-jour, de revenir à un backup, de consulter les logs et les comparaisons des mises-à-jour passées (cf. section précédente).

Actions:

- **status** : general status of the MocServer (size, existing backups, ...)
- **harvest** : launch the harvest process for populating the pilot directory (no change)
- **pilot** : compare the current MocServer content to the pilot files (no change)
- **validate** : replace data directory by pilot files and update MocServer content according to
- **update** : update the current MocServer content from the data directory

Liste exhaustive des paramètres d'interrogation

Version: v5.06 - May 2022

[Query parameters (.../query?xxx)]

help - This help. help=version => MocServer version number

RA - Input Cone Search right ascension in degrees ICRS (see IVOA CS REC)

DEC - Input Cone Search declination in degrees ICRS (see IVOA CS REC)

SR - Input Cone Search radius in degrees (see IVOA CS REC)

TIME - Input Cone Search time range as two MJD numbers (see IVOA CS REC&WD)

stc - Input STC polygon or circle region (see IVOA STC-S note)

moc - Input Inline SMOC, TMOC or STMOC (see IVOA MOC REC&WD)

spacesys - Input space system

timesys - Input time system

url - Input MOC url (FITS, ASCII or JSON syntax - see IVOA MOC REC&WD)

intersect - Input MOC intersection constraint: overlaps, enclosed, covers

expr - Input filter by expression (ex: obs_title=*NASA*,*CDS* &&

hips_service_url!=http://aladin*)

get - Output result: id, record, number, smoc, tmoc, stmoc, anymoc
(deprecated: moc, umoc, imoc)

fmt - Output format: ascii, json, fits, (deprecated: asciic, glu)

order - Output MOC order limit (ex1: t32 s13, ex2: 1MB)

op - Output MOC operation: union, intersection

fields - Output record fields: (ex: ID,obs_title,hips_service*

casesensitive - Output record fields case: true, false

MAXREC - Output record number

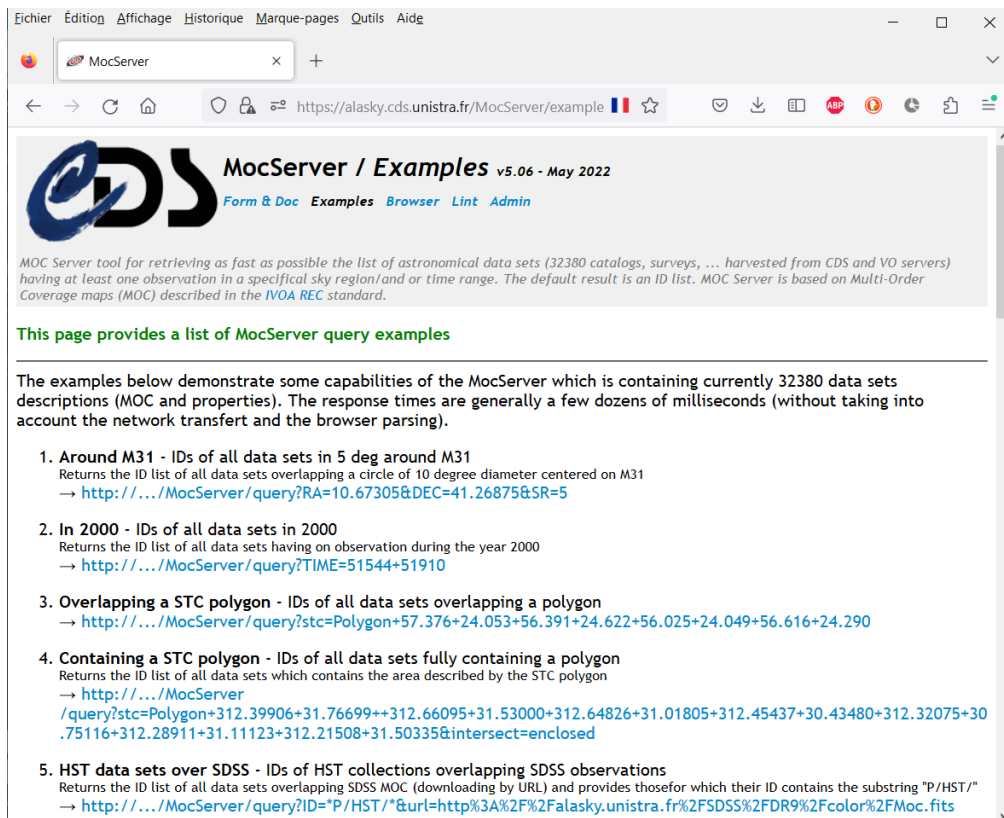
callback - [deprecated] Output callback JSON function

POS - [deprecated] => use Cone Search query

SIZE - [deprecated] => use Cone Search query

Table des matières

Introduction.....	3
Principe de fonctionnement.....	4
Sites Web.....	4
Origines des données	5
Propriétés des collections	5
Identifications des collections	6
Interrogations basiques.....	7
Interrogation par cône	7
Interrogation par polygone	7
Interrogation par intervalle temporel	8
Contraintes spatiales et temporelles	8
Interrogation par MOC.....	8
Interrogation par propriétés	10
Méthode basique par adjonction de paramètres à l'URL	10
Méthode avancée par expression algébrique	11
Résultat de la requête	12
Résultats sous forme de couverture MOC	13
Synchronisation des données pour un annuaire tiers.....	14
Utilisation du MocServer par Aladin Desktop	15
Liste exhaustive des paramètres d'interrogation	16



MocServer - Manuel de l'utilisateur
 Version de Mars 2023

© 2023 - Université de Strasbourg/CNRS – sous Licence Ouverte (compatible CC-BY)