Package 'paws.cost.management'

October 14, 2022

Title 'Amazon Web Services' Cost Management Services

Version 0.1.12

Description Interface to 'Amazon Web Services' cost management services, including cost and usage reports, budgets, pricing, and more <https://aws.amazon.com/>.

License Apache License (>= 2.0)

URL https://github.com/paws-r/paws

BugReports https://github.com/paws-r/paws/issues

Imports paws.common (>= 0.3.0)

Suggests testthat

Encoding UTF-8

RoxygenNote 7.1.1

Collate 'budgets_service.R' 'budgets_interfaces.R' 'budgets_operations.R' 'costandusagereportservice_service.R' 'costandusagereportservice_interfaces.R' 'costandusagereportservice_operations.R' 'costexplorer_service.R' 'costexplorer_interfaces.R' 'costexplorer_operations.R' 'marketplacecommerceanalytics_service.R' 'marketplacecommerceanalytics_interfaces.R' 'marketplacecommerceanalytics_operations.R' 'marketplaceentitlementservice_service.R' 'marketplaceentitlementservice_interfaces.R' 'marketplaceentitlementservice_operations.R' 'marketplacemetering_service.R' 'marketplacemetering_interfaces.R' 'marketplacemetering_operations.R' 'pricing_service.R' 'pricing_interfaces.R' 'pricing_operations.R'

NeedsCompilation no

Author David Kretch [aut, cre], Adam Banker [aut], Amazon.com, Inc. [cph]

Maintainer David Kretch <david.kretch@gmail.com>

Repository CRAN

Date/Publication 2021-08-23 07:11:00 UTC

R topics documented:

---

budgets *AWS Budgets*

Description

The AWS Budgets API enables you to use AWS Budgets to plan your service usage, service costs, and instance reservations. The API reference provides descriptions, syntax, and usage examples for each of the actions and data types for AWS Budgets.

Budgets provide you with a way to see the following information:

- How close your plan is to your budgeted amount or to the free tier limits
- Your usage-to-date, including how much you've used of your Reserved Instances (RIs)
- Your current estimated charges from AWS, and how much your predicted usage will accrue in charges by the end of the month
- How much of your budget has been used

AWS updates your budget status several times a day. Budgets track your unblended costs, subscriptions, refunds, and RIs. You can create the following types of budgets:

- **Cost budgets** - Plan how much you want to spend on a service.
- **Usage budgets** - Plan how much you want to use one or more services.
- **RI utilization budgets** - Define a utilization threshold, and receive alerts when your RI usage falls below that threshold. This lets you see if your RIs are unused or under-utilized.
- **RI coverage budgets** - Define a coverage threshold, and receive alerts when the number of your instance hours that are covered by RIs fall below that threshold. This lets you see how much of your instance usage is covered by a reservation.

Service Endpoint

The AWS Budgets API provides the following endpoint:

- https://budgets.amazonaws.com

For information about costs that are associated with the AWS Budgets API, see AWS Cost Management Pricing.

Usage

```
budgets(config = list())
```

Arguments

config	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- budgets(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

create_budget	Creates a budget and, if included, notifications and subscribers
create_budget_action	Creates a budget action
create_notification	Creates a notification
create_subscriber	Creates a subscriber
delete_budget	Deletes a budget
delete_budget_action	Deletes a budget action
delete_notification	Deletes a notification
delete_subscriber	Deletes a subscriber
describe_budget	Describes a budget
describe_budget_action	Describes a budget action detail
describe_budget_action_histories	Describes a budget action history detail

describe_budget_actions_for_account	Describes all of the budget actions for an account
describe_budget_actions_for_budget	Describes all of the budget actions for a budget
describe_budget_performance_history	Describes the history for DAILY, MONTHLY, and QUARTERLY budgets
describe_budgets	Lists the budgets that are associated with an account
describe_notifications_for_budget	Lists the notifications that are associated with a budget
describe_subscribers_for_notification	Lists the subscribers that are associated with a notification
execute_budget_action	Executes a budget action
update_budget	Updates a budget
update_budget_action	Updates a budget action
update_notification	Updates a notification
update_subscriber	Updates a subscriber

Examples

```
## Not run:
svc <- budgets()
svc$create_budget(
  Foo = 123
)

## End(Not run)
```

`costandusagereportservice` *AWS Cost and Usage Report Service*

Description

The AWS Cost and Usage Report API enables you to programmatically create, query, and delete AWS Cost and Usage report definitions.

AWS Cost and Usage reports track the monthly AWS costs and usage associated with your AWS account. The report contains line items for each unique combination of AWS product, usage type, and operation that your AWS account uses. You can configure the AWS Cost and Usage report to show only the data that you want, using the AWS Cost and Usage API.

Service Endpoint

The AWS Cost and Usage Report API provides the following endpoint:

- cur.us-east-1.amazonaws.com

Usage

```
costandusagereportservice(config = list())
```

Arguments

config	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- costandusagereportservice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

delete_report_definition	Deletes the specified report
describe_report_definitions	Lists the AWS Cost and Usage reports available to this account
modify_report_definition	Allows you to programatically update your report preferences
put_report_definition	Creates a new report using the description that you provide

Examples

```
## Not run:
svc <- costandusagereportservice()
# The following example deletes the AWS Cost and Usage report named
# ExampleReport.
svc$delete_report_definition(
  ReportName = "ExampleReport"
)

## End(Not run)
```

`costexplorer` *AWS Cost Explorer Service*

Description

The Cost Explorer API enables you to programmatically query your cost and usage data. You can query for aggregated data such as total monthly costs or total daily usage. You can also query for granular data, such as the number of daily write operations for Amazon DynamoDB database tables in your production environment.

Service Endpoint

The Cost Explorer API provides the following endpoint:

- `https://ce.us-east-1.amazonaws.com`

For information about costs associated with the Cost Explorer API, see AWS Cost Management Pricing.

Usage

```
costexplorer(config = list())
```

Arguments

`config`	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- costexplorer(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

create_anomaly_monitor	Creates a new cost anomaly detection monitor with the requested type and m
create_anomaly_subscription	Adds a subscription to a cost anomaly detection monitor
create_cost_category_definition	Creates a new Cost Category with the requested name and rules
delete_anomaly_monitor	Deletes a cost anomaly monitor
delete_anomaly_subscription	Deletes a cost anomaly subscription
delete_cost_category_definition	Deletes a Cost Category
describe_cost_category_definition	Returns the name, ARN, rules, definition, and effective dates of a Cost Categ
get_anomalies	Retrieves all of the cost anomalies detected on your account, during the time
get_anomaly_monitors	Retrieves the cost anomaly monitor definitions for your account
get_anomaly_subscriptions	Retrieves the cost anomaly subscription objects for your account
get_cost_and_usage	Retrieves cost and usage metrics for your account
get_cost_and_usage_with_resources	Retrieves cost and usage metrics with resources for your account
get_cost_categories	Retrieves an array of Cost Category names and values incurred cost
get_cost_forecast	Retrieves a forecast for how much Amazon Web Services predicts that you w
get_dimension_values	Retrieves all available filter values for a specified filter over a period of time
get_reservation_coverage	Retrieves the reservation coverage for your account
get_reservation_purchase_recommendation	Gets recommendations for which reservations to purchase
get_reservation_utilization	Retrieves the reservation utilization for your account
get_rightsizing_recommendation	Creates recommendations that help you save cost by identifying idle and und
get_savings_plans_coverage	Retrieves the Savings Plans covered for your account
get_savings_plans_purchase_recommendation	Retrieves your request parameters, Savings Plan Recommendations Summary
get_savings_plans_utilization	Retrieves the Savings Plans utilization for your account across date ranges w
get_savings_plans_utilization_details	Retrieves attribute data along with aggregate utilization and savings data for
get_tags	Queries for available tag keys and tag values for a specified period
get_usage_forecast	Retrieves a forecast for how much Amazon Web Services predicts that you w
list_cost_category_definitions	Returns the name, ARN, NumberOfRules and effective dates of all Cost Cate
provide_anomaly_feedback	Modifies the feedback property of a given cost anomaly
update_anomaly_monitor	Updates an existing cost anomaly monitor
update_anomaly_subscription	Updates an existing cost anomaly monitor subscription
update_cost_category_definition	Updates an existing Cost Category

Examples

```
## Not run:
svc <- costexplorer()
svc$create_anomaly_monitor(
  Foo = 123
)

## End(Not run)
```

`marketplacecommerceanalytics` *AWS Marketplace Commerce Analytics*

Description

Provides AWS Marketplace business intelligence data on-demand.

Usage

```
marketplacecommerceanalytics(config = list())
```

Arguments

`config`	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- marketplacecommerceanalytics(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

generate_data_set	Given a data set type and data set publication date, asynchronously publishes the requested data s
start_support_data_export	Given a data set type and a from date, asynchronously publishes the requested customer support d

Examples

```
## Not run:
svc <- marketplacecommerceanalytics()
svc$generate_data_set(
  Foo = 123
)

## End(Not run)
```

`marketplaceentitlementservice`
AWS Marketplace Entitlement Service

Description

This reference provides descriptions of the AWS Marketplace Entitlement Service API.

AWS Marketplace Entitlement Service is used to determine the entitlement of a customer to a given product. An entitlement represents capacity in a product owned by the customer. For example, a customer might own some number of users or seats in an SaaS application or some amount of data capacity in a multi-tenant database.

Getting Entitlement Records

- *GetEntitlements*- Gets the entitlements for a Marketplace product.

Usage

```
marketplaceentitlementservice(config = list())
```

Arguments

`config`	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- marketplaceentitlementservice(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
```

```
      session_token = "string"
    ),
    profile = "string"
  ),
  endpoint = "string",
  region = "string"
 )
)
```

Operations

get_entitlements	GetEntitlements retrieves entitlement values for a given product

Examples

```
## Not run:
svc <- marketplaceentitlementservice()
svc$get_entitlements(
  Foo = 123
)

## End(Not run)
```

`marketplacemetering` *AWSMarketplace Metering*

Description

AWS Marketplace Metering Service

This reference provides descriptions of the low-level AWS Marketplace Metering Service API.

AWS Marketplace sellers can use this API to submit usage data for custom usage dimensions.

For information on the permissions you need to use this API, see AWS Marketing metering and entitlement API permissions in the *AWS Marketplace Seller Guide.*

Submitting Metering Records

- *MeterUsage*- Submits the metering record for a Marketplace product. MeterUsage is called from an EC2 instance or a container running on EKS or ECS.
- *BatchMeterUsage*- Submits the metering record for a set of customers. BatchMeterUsage is called from a software-as-a-service (SaaS) application.

Accepting New Customers

- *ResolveCustomer*- Called by a SaaS application during the registration process. When a buyer visits your website during the registration process, the buyer submits a Registration Token through the browser. The Registration Token is resolved through this API to obtain a CustomerIdentifier and Product Code.

Entitlement and Metering for Paid Container Products

- Paid container software products sold through AWS Marketplace must integrate with the AWS Marketplace Metering Service and call the RegisterUsage operation for software entitlement and metering. Free and BYOL products for Amazon ECS or Amazon EKS aren't required to call RegisterUsage, but you can do so if you want to receive usage data in your seller reports. For more information on using the RegisterUsage operation, see Container-Based Products.

BatchMeterUsage API calls are captured by AWS CloudTrail. You can use Cloudtrail to verify that the SaaS metering records that you sent are accurate by searching for records with the eventName of BatchMeterUsage. You can also use CloudTrail to audit records over time. For more information, see the *AWS CloudTrail User Guide* .

Usage

```
marketplacemetering(config = list())
```

Arguments

`config`	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- marketplacemetering(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

batch_meter_usage	BatchMeterUsage is called from a SaaS application listed on the AWS Marketplace to post metering reco
meter_usage	API to emit metering records
register_usage	Paid container software products sold through AWS Marketplace must integrate with the AWS Marketpl
resolve_customer	ResolveCustomer is called by a SaaS application during the registration process

Examples

```
## Not run:
svc <- marketplacemetering()
svc$batch_meter_usage(
  Foo = 123
)

## End(Not run)
```

`pricing` *AWS Price List Service*

Description

AWS Price List Service API (AWS Price List Service) is a centralized and convenient way to programmatically query Amazon Web Services for services, products, and pricing information. The AWS Price List Service uses standardized product attributes such as `Location`, `Storage Class`, and `Operating System`, and provides prices at the SKU level. You can use the AWS Price List Service to build cost control and scenario planning tools, reconcile billing data, forecast future spend for budgeting purposes, and provide cost benefit analysis that compare your internal workloads with AWS.

Use `GetServices` without a service code to retrieve the service codes for all AWS services, then `GetServices` with a service code to retreive the attribute names for that service. After you have the service code and attribute names, you can use `get_attribute_values` to see what values are available for an attribute. With the service code and an attribute name and value, you can use `get_products` to find specific products that you're interested in, such as an `AmazonEC2` instance, with a `Provisioned IOPS` `volumeType`.

Service Endpoint

AWS Price List Service API provides the following two endpoints:

- https://api.pricing.us-east-1.amazonaws.com
- https://api.pricing.ap-south-1.amazonaws.com

Usage

```
pricing(config = list())
```

Arguments

config	Optional configuration of credentials, endpoint, and/or region.

Value

A client for the service. You can call the service's operations using syntax like `svc$operation(...)`, where `svc` is the name you've assigned to the client. The available operations are listed in the Operations section.

Service syntax

```
svc <- pricing(
  config = list(
    credentials = list(
      creds = list(
        access_key_id = "string",
        secret_access_key = "string",
        session_token = "string"
      ),
      profile = "string"
    ),
    endpoint = "string",
    region = "string"
  )
)
```

Operations

describe_services	Returns the metadata for one service or a list of the metadata for all services
get_attribute_values	Returns a list of attribute values
get_products	Returns a list of all products that match the filter criteria

Examples

```
## Not run:
svc <- pricing()
svc$describe_services(
  FormatVersion = "aws_v1",
  MaxResults = 1L,
  ServiceCode = "AmazonEC2"
)

## End(Not run)
```

Index