#@namespace java com.twitter.unified\_user\_actions.thriftjava

#@namespace scala com.twitter.unified\_user\_actions.thriftscala

#@namespace strato com.twitter.unified\_user\_actions

include "com/twitter/unified\_user\_actions/metadata.thrift"

include "com/twitter/search/common/constants/query.thrift"

include "com/twitter/search/common/constants/result.thrift"

/\*

\* Represents the product surface on which an action took place.

\* See reference that delineates various product surfaces:

\* https://docs.google.com/document/d/1PS2ZOyNUoUdO45zxhE7dH3L8KUcqJwo6Vx-XUGGFo6U

\* Note: the implementation here may not reflect the above doc exactly.

\*/

enum ProductSurface {

// 1 - 19 for Home

HomeTimeline = 1

// 20 - 39 for Notifications

NotificationTab = 20

PushNotification = 21

EmailNotification = 22

// 40 - 59 for Search

SearchResultsPage = 40

SearchTypeahead = 41

// 60 - 79 for Tweet Details Page (Conversation Page)

TweetDetailsPage = 60

// 80 - 99 for Profile Page

ProfilePage = 80

// 100 - 119 for ?

RESERVED\_100 = 100

// 120 - 139 for ?

RESERVED\_120 = 120

}(persisted='true', hasPersonalData='false')

union ProductSurfaceInfo {

// 1 matches the enum index HomeTimeline in ProductSurface

1: HomeTimelineInfo homeTimelineInfo

// 20 matches the enum index NotificationTab in ProductSurface

20: NotificationTabInfo notificationTabInfo

// 21 matches the enum index PushNotification in ProductSurface

21: PushNotificationInfo pushNotificationInfo

// 22 matches the enum index EmailNotification in ProductSurface

22: EmailNotificationInfo emailNotificationInfo

// 40 matches the enum index SearchResultPage in ProductSurface

40: SearchResultsPageInfo searchResultsPageInfo

// 41 matches the enum index SearchTypeahead in ProductSurface

41: SearchTypeaheadInfo searchTypeaheadInfo

// 60 matches the enum index TweetDetailsPage in ProductSurface

60: TweetDetailsPageInfo tweetDetailsPageInfo

// 80 matches the enum index ProfilePage in ProductSurface

80: ProfilePageInfo profilePageInfo

}(persisted='true', hasPersonalData='false')

/\*

\* Please keep this minimal to avoid overhead. It should only

\* contain high value Home Timeline specific attributes.

\*/

struct HomeTimelineInfo {

// suggestType is deprecated, please do't re-use!

// 1: optional i32 suggestType

2: optional string suggestionType

3: optional i32 injectedPosition

}(persisted='true', hasPersonalData='false')

struct NotificationTabInfo {

/\*

\* Note that this field represents the `impressionId` in a Notification Tab notification.

\* It has been renamed to `notificationId` in UUA so that the name effectively represents the

\* value it holds, i.e., a unique id for a notification and request.

\*/

1: required string notificationId(personalDataType='UniversallyUniqueIdentifierUuid')

}(persisted='true', hasPersonalData='false')

struct PushNotificationInfo {

/\*

\* Note that this field represents the `impressionId` in a Push Notification.

\* It has been renamed to `notificationId` in UUA so that the name effectively represents the

\* value it holds, i.e., a unique id for a notification and request.

\*/

1: required string notificationId(personalDataType='UniversallyUniqueIdentifierUuid')

}(persisted='true', hasPersonalData='false')

struct EmailNotificationInfo {

/\*

\* Note that this field represents the `impressionId` in an Email Notification.

\* It has been renamed to `notificationId` in UUA so that the name effectively represents the

\* value it holds, i.e., a unique id for a notification and request.

\*/

1: required string notificationId(personalDataType='UniversallyUniqueIdentifierUuid')

}(persisted='true', hasPersonalData='false')

struct TweetDetailsPageInfo {

// To be deprecated, please don't re-use!

// Only reason to keep it now is Sparrow doesn't take empty struct. Once there is a real

// field we should just comment it out.

1: required list<string> breadcrumbViews(personalDataType = 'WebsitePage')

// Deprecated, please don't re-use!

// 2: required list<metadata.BreadcrumbTweet> breadcrumbTweets(personalDataType = 'TweetId')

}(persisted='true', hasPersonalData='true')

struct ProfilePageInfo {

// To be deprecated, please don't re-use!

// Only reason to keep it now is Sparrow doesn't take empty struct. Once there is a real

// field we should just comment it out.

1: required list<string> breadcrumbViews(personalDataType = 'WebsitePage')

// Deprecated, please don't re-use!

// 2: required list<metadata.BreadcrumbTweet> breadcrumbTweets(personalDataType = 'TweetId')

}(persisted='true', hasPersonalData='true')

struct SearchResultsPageInfo {

// search query string

1: required string query(personalDataType = 'SearchQuery')

// Attribution of the search (e.g. Typed Query, Hashtag Click, etc.)

// see http://go/sgb/src/thrift/com/twitter/search/common/constants/query.thrift for details

2: optional query.ThriftQuerySource querySource

// 0-indexed position of item in list of search results

3: optional i32 itemPosition

// Attribution of the tweet result (e.g. QIG, Earlybird, etc)

// see http://go/sgb/src/thrift/com/twitter/search/common/constants/result.thrift for details

4: optional set<result.TweetResultSource> tweetResultSources

// Attribution of the user result (e.g. ExpertSearch, QIG, etc)

// see http://go/sgb/src/thrift/com/twitter/search/common/constants/result.thrift for details

5: optional set<result.UserResultSource> userResultSources

// The query filter type on the Search Results Page (SRP) when the action took place.

// Clicking on a tab in SRP applies a query filter automatically.

6: optional SearchQueryFilterType queryFilterType

}(persisted='true', hasPersonalData='true')

struct SearchTypeaheadInfo {

// search query string

1: required string query(personalDataType = 'SearchQuery')

// 0-indexed position of item in list of typeahead drop-down

2: optional i32 itemPosition

}(persisted='true', hasPersonalData='true')

enum SearchQueryFilterType {

// filter to top ranked content for a query

TOP = 1

// filter to latest content for a query

LATEST = 2

// filter to user results for a query

PEOPLE = 3

// filter to photo tweet results for a query

PHOTOS = 4

// filter to video tweet results for a query

VIDEOS = 5

}