» Data Source: azuread_application

Use this data source to access information about an existing Application within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write all (or owned by) applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API.

» Example Usage

```
data "azuread_application" "example" {
  name = "My First AzureAD Application"
}

output "azure_ad_object_id" {
  value = "${data.azuread_application.example.id}"
}
```

» Argument Reference

- `object_id` - (Optional) Specifies the Object ID of the Application within Azure Active Directory.
- `name` - (Optional) Specifies the name of the Application within Azure Active Directory.

NOTE: Either an `object_id` or `name` must be specified.

» Attributes Reference

- `id` - the Object ID of the Azure Active Directory Application.
- `application_id` - the Application ID of the Azure Active Directory Application.
- `available_to_other_tenants` - Is this Azure AD Application available to other tenants?
- `identifier_uris` - A list of user-defined URI(s) that uniquely identify a Web application within it's Azure AD tenant, or within a verified custom domain if the application is multi-tenant.
- `oauth2_allow_implicit_flow` - Does this Azure AD Application allow OAuth2.0 implicit flow tokens?

- `object_id` - the Object ID of the Azure Active Directory Application.
- `reply_urls` - A list of URLs that user tokens are sent to for sign in, or the redirect URIs that OAuth 2.0 authorization codes and access tokens are sent to.
- `group_membership_claims` - The `groups` claim issued in a user or OAuth 2.0 access token that the app expects.
- `required_resource_access` - A collection of `required_resource_access` blocks as documented below.
- `oauth2_permissions` - A collection of OAuth 2.0 permission scopes that the web API (resource) app exposes to client apps. Each permission is covered by a `oauth2_permission` block as documented below.
- `app_roles` - A collection of `app_role` blocks as documented below. For more information https://docs.microsoft.com/en-us/azure/architecture/multitenant-identity/app-roles

---

`required_resource_access` block exports the following:

- `resource_app_id` - The unique identifier for the resource that the application requires access to.
- `resource_access` - A collection of `resource_access` blocks as documented below

`resource_access` block exports the following:

- `id` - The unique identifier for one of the `OAuth2Permission` or `AppRole` instances that the resource application exposes.
- `type` - Specifies whether the id property references an `OAuth2Permission` or an `AppRole`.

`oauth2_permission` block exports the following:

- `id` - The unique identifier for one of the `OAuth2Permission`
- `type` - The type of the permission
- `admin_consent_description` - The description of the admin consent
- `admin_consent_display_name` - The display name of the admin consent
- `is_enabled` - Is this permission enabled?
- `user_consent_description` - The description of the user consent
- `user_consent_display_name` - The display name of the user consent

- `value` - The name of this permission

`app_role` block exports the following:

- `id` - The unique identifier of the `app_role`.
- `allowed_member_types` - Specifies whether this app role definition can be assigned to users and groups, or to other applications (that are accessing this application in daemon service scenarios). Possible values are: `User` and `Application`, or both.
- `description` - Permission help text that appears in the admin app assignment and consent experiences.
- `display_name` - Display name for the permission that appears in the admin consent and app assignment experiences.
- `is_enabled` - Determines if the app role is enabled.
- `value` - Specifies the value of the roles claim that the application should expect in the authentication and access tokens.

» Data Source: azuread_domains

Use this data source to access information about an existing Domains within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Directory.Read.All` within the `Windows Azure Active Directory` API.

» Example Usage

```
data "azuread_domains" "aad_domains" {}

output "domains" {
  value = "${data.azuread_domains.aad_domains.domains}"
}
```

» Argument Reference

- `include_unverified` - (Optional) Set to `true` if unverified Azure AD Domains should be included. Defaults to `false`.
- `only_default` - (Optional) Set to `true` to only return the default domain.

- `only_initial` - (Optional) Set to `true` to only return the initial domain, which is your primary Azure Active Directory tenant domain. Defaults to `false`.

NOTE: If `include_unverified` is set to `true` you cannot specify `only_default` or `only_initial`. Additionally you cannot combine `only_default` with `only_initial`.

» Attributes Reference

- `domains` - One or more `domain` blocks as defined below.

The `domain` block contains:

- `domain_name` - The name of the domain.
- `authentication_type` - The authentication type of the domain (Managed or Federated).
- `is_default` - `True` if this is the default domain that is used for user creation.
- `is_initial` - `True` if this is the initial domain created by Azure Activie Directory.
- `is_verified` - `True` if the domain has completed domain ownership verification.

» Data Source: azuread_group

Gets information about an Azure Active Directory group.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Read directory data` within the `Windows Azure Active Directory` API.

» Example Usage (by Group Display Name)

```
data "azuread_group" "example" {
  name = "A-AD-Group"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Optional) The Name of the AD Group we want to lookup.

- `object_id` - (Optional) Specifies the Object ID of the AD Group within Azure Active Directory.

NOTE: Either a `name` or an `object_id` must be specified.

» Attributes Reference

The following attributes are exported:

- `id` - The Object ID of the Azure AD Group.

» Data Source: azuread_service_principal

Gets information about an existing Service Principal associated with an Application within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write all applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API.

» Example Usage (by Application Display Name)

```
data "azuread_service_principal" "example" {
  display_name = "my-awesome-application"
}
```

» Example Usage (by Application ID)

```
data "azuread_service_principal" "example" {
  application_id = "00000000-0000-0000-0000-000000000000"
}
```

» Example Usage (by Object ID)

```
data "azuread_service_principal" "example" {
  object_id = "00000000-0000-0000-0000-000000000000"
}
```

» Argument Reference

The following arguments are supported:

- `application_id` - (Optional) The ID of the Azure AD Application.
- `object_id` - (Optional) The ID of the Azure AD Service Principal.
- `display_name` - (Optional) The Display Name of the Azure AD Application associated with this Service Principal.

NOTE: At least one of `application_id`, `display_name` or `object_id` must be specified.

- `app_roles` - A collection of `app_role` blocks as documented below. For more information https://docs.microsoft.com/en-us/azure/architecture/multitenant-identity/app-roles
- `oauth2_permissions` - A collection of OAuth 2.0 permissions exposed by the associated application. Each permission is covered by a `oauth2_permission` block as documented below.

» Attributes Reference

The following attributes are exported:

- `id` - The Object ID for the Service Principal.

`oauth2_permission` block exports the following:

- `id` - The unique identifier for one of the `OAuth2Permission`
- `type` - The type of the permission
- `admin_consent_description` - The description of the admin consent
- `admin_consent_display_name` - The display name of the admin consent
- `is_enabled` - Is this permission enabled?
- `user_consent_description` - The description of the user consent
- `user_consent_display_name` - The display name of the user consent
- `value` - The name of this permission

`app_role` block exports the following:

- `id` - The unique identifier of the `app_role`.

- `allowed_member_types` - Specifies whether this app role definition can be assigned to users and groups, or to other applications (that are accessing this application in daemon service scenarios). Possible values are: `User` and `Application`, or both.
- `description` - Permission help text that appears in the admin app assignment and consent experiences.
- `display_name` - Display name for the permission that appears in the admin consent and app assignment experiences.
- `is_enabled` - Determines if the app role is enabled.
- `value` - Specifies the value of the roles claim that the application should expect in the authentication and access tokens.

» Data Source: azuread_user

Gets information about an Azure Active Directory user.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Read directory data` within the `Windows Azure Active Directory` API.

» Example Usage

```
data "azuread_user" "example" {
  user_principal_name = "user@hashicorp.com"
}
```

» Argument Reference

The following arguments are supported:

- `user_principal_name` - (Required) The User Principal Name of the Azure AD User.
- `object_id` - (Optional) Specifies the Object ID of the Application within Azure Active Directory.

NOTE: Either a `user_principal_name` or an `object_id` must be specified.

» Attributes Reference

The following attributes are exported:

- `id` - The Object ID of the Azure AD User.
- `user_principal_name` - The User Principal Name of the Azure AD User.
- `account_enabled` - `True` if the account is enabled; otherwise `False`.
- `display_name` - The Display Name of the Azure AD User.
- `mail` - The primary email address of the Azure AD User.
- `mail_nickname` - The email alias of the Azure AD User.

» Data Source: azuread_user

Gets Object IDs or UPNs for multiple Azure Active Directory users.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Read directory data` within the `Windows Azure Active Directory` API.

» Example Usage

```
data "azuread_users" "users" {
  user_principal_names = ["kat@hashicorp.com", "byte@hashicorp.com"]
}
```

» Argument Reference

The following arguments are supported:

- `user_principal_names` - (optional) The User Principal Names of the Azure AD Users.
- `object_ids` - (Optional) The Object IDs of the Azure AD Users.

NOTE: Either `user_principal_names` or `object_ids` must be specified.

» Attributes Reference

The following attributes are exported:

- `object_ids` - The Object IDs of the Azure AD Users.
- `user_principal_names` - The User Principal Names of the Azure AD Users.

» azuread_application

Manages an Application within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write owned by applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API.

» Example Usage

```
resource "azuread_application" "example" {
  name                       = "example"
  homepage                   = "https://homepage"
  identifier_uris            = ["https://uri"]
  reply_urls                 = ["https://replyurl"]
  available_to_other_tenants = false
  oauth2_allow_implicit_flow = true
  type                       = "webapp/api"

  required_resource_access {
    resource_app_id = "00000003-0000-0000-c000-000000000000"

    resource_access {
      id   = "..."
      type = "Role"
    }

    resource_access {
      id   = "..."
      type = "Scope"
    }

    resource_access {
      id   = "..."
      type = "Scope"
    }
  }

  required_resource_access {
    resource_app_id = "00000002-0000-0000-c000-000000000000"

    resource_access {
      id   = "..."
      type = "Scope"
    }
```

```
  }

  app_role {
    allowed_member_types = [
      "User",
      "Application",
    ]

    description  = "Admins can manage roles and perform all task actions"
    display_name = "Admin"
    is_enabled   = true
    value        = "Admin"
  }
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) The display name for the application.
- `homepage` - (optional) The URL to the application's home page. If no homepage is specified this defaults to `https://{name}`.
- `identifier_uris` - (Optional) A list of user-defined URI(s) that uniquely identify a Web application within it's Azure AD tenant, or within a verified custom domain if the application is multi-tenant.
- `reply_urls` - (Optional) A list of URLs that user tokens are sent to for sign in, or the redirect URIs that OAuth 2.0 authorization codes and access tokens are sent to.
- `available_to_other_tenants` - (Optional) Is this Azure AD Application available to other tenants? Defaults to `false`.
- `public_client` - (Optional) Is this Azure AD Application a public client? Defaults to `false`.
- `oauth2_allow_implicit_flow` - (Optional) Does this Azure AD Application allow OAuth2.0 implicit flow tokens? Defaults to `false`.
- `group_membership_claims` - (Optional) Configures the `groups` claim issued in a user or OAuth 2.0 access token that the app expects. Defaults to `SecurityGroup`. Possible values are `None`, `SecurityGroup` or `All`.
- `required_resource_access` - (Optional) A collection of `required_resource_access` blocks as documented below.

- `type` - (Optional) Type of an application: `webapp/api` or `native`. Defaults to `webapp/api`. For `native` apps type `identifier_uris` property can not not be set.
- `app_role` - (Optional) A collection of `app_role` blocks as documented below. For more information https://docs.microsoft.com/en-us/azure/architecture/multitenant-identity/app-roles

---

`required_resource_access` supports the following:

- `resource_app_id` - (Required) The unique identifier for the resource that the application requires access to. This should be equal to the appId declared on the target resource application.
- `resource_access` - (Required) A collection of `resource_access` blocks as documented below.

`resource_access` supports the following:

- `id` - (Required) The unique identifier for one of the `OAuth2Permission` or `AppRole` instances that the resource application exposes.
- `type` - (Required) Specifies whether the id property references an `OAuth2Permission` or an `AppRole`. Possible values are `Scope` or `Role`.

`app_role` supports the following:

- `id` - The unique identifier of the `app_role`.
- `allowed_member_types` - (Required) Specifies whether this app role definition can be assigned to users and groups by setting to `User`, or to other applications (that are accessing this application in daemon service scenarios) by setting to `Application`, or to both.
- `description` - (Required) Permission help text that appears in the admin app assignment and consent experiences.
- `display_name` - (Required) Display name for the permission that appears in the admin consent and app assignment experiences.
- `is_enabled` - (Optional) Determines if the app role is enabled: Defaults to `true`.
- `value` - (Required) Specifies the value of the roles claim that the application should expect in the authentication and access tokens.

» Attributes Reference

The following attributes are exported:

- `application_id` - The Application ID.
- `object_id` - The Application's Object ID.
- `oauth2_permissions` - A collection of OAuth 2.0 permission scopes that the web API (resource) app exposes to client apps. Each permission is covered by a `oauth2_permission` block as documented below.

`oauth2_permission` block exports the following:

- `id` - The unique identifier for one of the `OAuth2Permission`.
- `type` - The type of the permission.
- `admin_consent_description` - The description of the admin consent.
- `admin_consent_display_name` - The display name of the admin consent.
- `is_enabled` - Is this permission enabled?
- `user_consent_description` - The description of the user consent.
- `user_consent_display_name` - The display name of the user consent.
- `value` - The name of this permission.

» Import

Azure Active Directory Applications can be imported using the `object id`, e.g.

```
terraform import azuread_application.test 00000000-0000-0000-0000-000000000000
```

» azuread_application_password

Manages a Password associated with an Application within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write all applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API.

» Example Usage

```
resource "azuread_application" "example" {
  name                       = "example"
  homepage                   = "http://homepage"
  identifier_uris            = ["http://uri"]
  reply_urls                 = ["http://replyurl"]
  available_to_other_tenants = false
  oauth2_allow_implicit_flow = true
}

resource "azuread_application_password" "example" {
  application_id = "${azuread_application.example.id}"
  value          = "VT=uSgbTanZhyz@%nL9Hpd+Tfay_MRV#"
  end_date       = "2020-01-01T01:02:03Z"
}
```

» Argument Reference

The following arguments are supported:

- `application_object_id` - (Required) The Object ID of the Application for which this password should be created. Changing this field forces a new resource to be created.
- `value` - (Required) The Password for this Application .
- `end_date` - (Optional) The End Date which the Password is valid until, formatted as a RFC3339 date string (e.g. `2018-01-01T01:02:03Z`). Changing this field forces a new resource to be created.
- `end_date_relative` - (Optional) A relative duration for which the Password is valid until, for example `240h` (10 days) or `2400h30m`. Changing this field forces a new resource to be created.

NOTE: One of `end_date` or `end_date_relative` must be set.

- `key_id` - (Optional) A GUID used to uniquely identify this Password. If not specified a GUID will be created. Changing this field forces a new resource to be created.
- `start_date` - (Optional) The Start Date which the Password is valid from, formatted as a RFC3339 date string (e.g. `2018-01-01T01:02:03Z`). If this isn't specified, the current date is used. Changing this field forces a new resource to be created.

» Attributes Reference

The following attributes are exported:

- `id` - The Key ID for the Password.

» Import

Passwords can be imported using the `object id` of an Application, e.g.

```
terraform import azuread_application_password.test 00000000-0000-0000-0000-000000000000/1111
```

NOTE: This ID format is unique to Terraform and is composed of the Application's Object ID and the Password's Key ID in the format `{ObjectId}/{PasswordKeyId}`.

» azuread_group

Manages a Group within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Read and write all groups` within the `Windows Azure Active Directory` API. In addition it must also have either the `Company Administrator` or `User Account Administrator` Azure Active Directory roles assigned in order to be able to delete groups. You can assign one of the required Azure Active Directory Roles with the **AzureAD PowerShell Module**, which is available for Windows PowerShell or in the Azure Cloud Shell. Please refer to this documentation for more details.

» Example Usage

Basic example

```
resource "azuread_group" "example" {
  name = "A-AD-Group"
}
```

A group with members

```
resource "azuread_user" "example" {
  display_name          = "J Doe"
  password              = "notSecure123"
  user_principal_name   = "j.doe@terraform.onmicrosoft.com"
}

resource "azuread_group" "example" {
```

```
  name    = "MyGroup"
  members = [ "${azuread_user.example.object_id}" /*, more users */ ]
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) The display name for the Group. Changing this forces a new resource to be created.
- `members` (Optional) A set of members who should be present in this Group. Supported Object types are Users, Groups or Service Principals.
- `owners` (Optional) A set of owners who own this Group. Supported Object types are Users or Service Principals.

NOTE: Group names are not unique within Azure Active Directory.

NOTE: Do not use `azuread_group_member` at the same time as the `members` argument.

NOTE: Do not use `azuread_group_owner` at the same time as the `owners` argument.

» Attributes Reference

The following attributes are exported:

- `id` - The Object ID of the Group.
- `name` - The Display Name of the Group.
- `members` - The Members of the Group.
- `owners` - The Members of the Group.

» Import

Azure Active Directory Groups can be imported using the `object id`, e.g.

```
terraform import azuread_group.my_group 00000000-0000-0000-0000-000000000000
```

» azuread_group_member

Manages a single Group Membership within Azure Active Directory.

NOTE: Do not use this resource at the same time as `azuread_group.members`.

» Example Usage

```
data "azuread_user" "example" {
  user_principal_name = "jdoe@hashicorp.com"
}

resource "azuread_group" "example" {
  name = "my_group"
}

resource "azuread_group_member" "example" {
  group_object_id   = "${azuread_group.example.id}"
  member_object_id  = "${data.azuread_user.example.id}"
}
```

» Argument Reference

The following arguments are supported:

- `group_object_id` - (Required) The Object ID of the Azure AD Group you want to add the Member to. Changing this forces a new resource to be created.
- `member_object_id` - (Required) The Object ID of the Azure AD Object you want to add as a Member to the Group. Supported Object types are Users, Groups or Service Principals. Changing this forces a new resource to be created.

NOTE: The Member object has to be present in your Azure Active Directory, either as a Member or a Guest.

» Attributes Reference

The following attributes are exported:

- `id` - The ID of the Azure AD Group Member.

» Import

Azure Active Directory Group Members can be imported using the `object id`, e.g.

```
terraform import azuread_group_member.test 00000000-0000-0000-0000-000000000000/11111111-111
```

NOTE: This ID format is unique to Terraform and is composed of the Azure AD Group Object ID and the target Member Object ID in the format `{GroupObjectID}/{MemberObjectID}`.

» azuread_service_principal

Manages a Service Principal associated with an Application within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write all applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API. Please see The Granting a Service Principal permission to manage AAD for the required steps.

» Example Usage

```
resource "azuread_application" "example" {
  name                       = "example"
  homepage                   = "http://homepage"
  identifier_uris            = ["http://uri"]
  reply_urls                 = ["http://replyurl"]
  available_to_other_tenants = false
  oauth2_allow_implicit_flow = true
}

resource "azuread_service_principal" "example" {
  application_id               = "${azuread_application.example.application_id}"
  app_role_assignment_required  = false

  tags = ["example", "tags", "here"]
}
```

» Argument Reference

The following arguments are supported:

- `application_id` - (Required) The ID of the Azure AD Application for which to create a Service Principal.
- `app_role_assignment_required` - (Optional) Does this Service Principal require an AppRoleAssignment to a user or group before Azure AD will issue a user or access token to the application? Defaults to `false`.
- `tags` - (Optional) A list of tags to apply to the Service Principal.

» Attributes Reference

The following attributes are exported:

- `id` - The Object ID (internal ID) for the Service Principal.
- `application_id` - The Application ID (appId) for the Service Principal.
- `object_id` - The Service Principal's Object ID.
- `display_name` - The Display Name of the Azure Active Directory Application associated with this Service Principal.
- `app_role_assignment_required` - Whether this Service Principal requires an AppRoleAssignment to a user or group before Azure AD will issue a user or access token to the application.
- `oauth2_permissions` - A collection of OAuth 2.0 permissions exposed by the associated application. Each permission is covered by a `oauth2_permission` block as documented below.

`oauth2_permission` block exports the following:

- `id` - The unique identifier for one of the `OAuth2Permission`.
- `type` - The type of the permission.
- `admin_consent_description` - The description of the admin consent.
- `admin_consent_display_name` - The display name of the admin consent.
- `is_enabled` - Is this permission enabled?
- `user_consent_description` - The description of the user consent.
- `user_consent_display_name` - The display name of the user consent.
- `value` - The name of this permission.

» Import

Azure Active Directory Service Principals can be imported using the `object id`, e.g.

```
terraform import azuread_service_principal.test 00000000-0000-0000-0000-000000000000
```

» azuread_service_principal_password

Manages a Password associated with a Service Principal within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to both `Read and write all applications` and `Sign in and read user profile` within the `Windows Azure Active Directory` API.

» Example Usage

```
resource "azuread_application" "example" {
  name                       = "example"
  homepage                   = "http://homepage"
  identifier_uris            = ["http://uri"]
  reply_urls                 = ["http://replyurl"]
  available_to_other_tenants = false
  oauth2_allow_implicit_flow = true
}

resource "azuread_service_principal" "example" {
  application_id = "${azuread_application.example.application_id}"
}

resource "azuread_service_principal_password" "example" {
  service_principal_id = "${azuread_service_principal.test.id}"
  value                = "VT=uSgbTanZhyz@%nL9Hpd+Tfay_MRV#"
  end_date             = "2020-01-01T01:02:03Z"
}
```

» Argument Reference

The following arguments are supported:

- `service_principal_id` - (Required) The ID of the Service Principal for which this password should be created. Changing this field forces a new resource to be created.
- `value` - (Required) The Password for this Service Principal.
- `end_date` - (Optional) The End Date which the Password is valid until, formatted as a RFC3339 date string (e.g. `2018-01-01T01:02:03Z`). Changing this field forces a new resource to be created.
- `end_date_relative` - (Optional) A relative duration for which the Password is valid until, for example `240h` (10 days) or `2400h30m`. Changing this field forces a new resource to be created.

NOTE: One of `end_date` or `end_date_relative` must be set.

- `key_id` - (Optional) A GUID used to uniquely identify this Key. If not specified a GUID will be created. Changing this field forces a new resource to be created.
- `start_date` - (Optional) The Start Date which the Password is valid from, formatted as a RFC3339 date string (e.g. `2018-01-01T01:02:03Z`). If this

isn't specified, the current date is used. Changing this field forces a new resource to be created.

» Attributes Reference

The following attributes are exported:

- `id` - The Key ID for the Service Principal Password.

» Import

Service Principal Passwords can be imported using the `object id`, e.g.

```
terraform import azuread_service_principal_password.test 00000000-0000-0000-0000-0000000000
```

NOTE: This ID format is unique to Terraform and is composed of the Service Principal's Object ID and the Service Principal Password's Key ID in the format `{ServicePrincipalObjectId}/{ServicePrincipalPasswordKeyId}`.

» azuread_user

Manages a User within Azure Active Directory.

NOTE: If you're authenticating using a Service Principal then it must have permissions to `Directory.ReadWrite.All` within the `Windows Azure Active Directory` API.

» Example Usage

```
resource "azuread_user" "example" {
  user_principal_name = "jdo@hashicorp.com"
  display_name        = "J. Doe"
  mail_nickname       = "jdoe"
  password            = "SecretP@sswd99!"
}
```

» Argument Reference

The following arguments are supported:

- `user_principal_name` - (Required) The User Principal Name of the Azure AD User.

- `display_name` - (Required) The name to display in the address book for the user.
- `account_enabled` - (Optional) `true` if the account should be enabled, otherwise `false`. Defaults to `true`.
- `mail_nickname`- (Optional) The mail alias for the user. Defaults to the user name part of the User Principal Name.
- `password` - (Required) The password for the User. The password must satisfy minimum requirements as specified by the password policy. The maximum length is 256 characters.
- `force_password_change` - (Optional) `true` if the User is forced to change the password during the next sign-in. Defaults to `false`.

» Attributes Reference

The following attributes are exported:

- `object_id` - The Object ID of the Azure AD User.
- `id` - The Object ID of the Azure AD User.
- `mail` - The primary email address of the Azure AD User.

» Import

Azure Active Directory Users can be imported using the `object id`, e.g.

```
terraform import azuread_user.my_user 00000000-0000-0000-0000-000000000000
```