» tfe_organization

Manages organizations.

» Example Usage

Basic usage:

```
resource "tfe_organization" "organization" {
  name = "my-org-name"
  email = "admin@company.com"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of the organization.
- `email` - (Required) Admin email address.
- `session_timeout_minutes` - (Optional) Session timeout after inactivity. Defaults to `20160`.
- `session_remember_minutes` - (Optional) Session expiration. Defaults to `20160`.
- `collaborator_auth_policy` - (Optional) Authentication policy (`password` or `two_factor_mandatory`). Defaults to `password`.

» Attributes Reference

- `id` - The name of the organization.

» tfe_organization_token

Generates a new organization token, replacing any existing token. This token can be used to act as the organization service account.

» Example Usage

Basic usage:

```
resource "tfe_organization_token" "token" {
  organization = "my-org-name"
}
```

» Argument Reference

The following arguments are supported:

- `organization` - (Required) Name of the organization.
- `force_regenerate` - (Optional) If set to `true`, a new token will be generated even if a token already exists. This will invalidate the existing token!

» Attributes Reference

- `id` - The ID of the token.
- `token` - The generated token.

» tfe_sentinel_policy

Sentinel Policy as Code is an embedded policy as code framework integrated with Terraform Enterprise.

Policies are configured on a per-organization level, and are enforced on all of an organization's workspaces during runs. Each plan's changes are validated against the policy prior to the apply step.

» Example Usage

Basic usage:

```
resource "tfe_sentinel_policy" "policy" {
  name = "my-policy-name"
  organization = "my-org-name"
  policy = "main = rule { true }"
  enforce_mode = "hard-mandatory"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of the policy.
- `organization` - (Required) Name of the organization.
- `policy` - (Required) The actual policy itself.

- `enforce_mode` - (Required) The enforcement level of the policy. Valid values are `advisory`, `hard-mandatory` and `soft-mandatory`. Defaults to `soft-mandatory`.

» Attributes Reference

- `id` - The ID of the policy.

» tfe_ssh_key

This resource represents an SSH key which includes a name and the SSH private key. An organization can have multiple SSH keys available.

» Example Usage

Basic usage:

```
resource "tfe_ssh_key" "ssh-key" {
  name = "my-ssh-key-name"
  organization = "my-org-name"
  key = "private-ssh-key"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name to identify the SSH key.
- `organization` - (Required) Name of the organization.
- `key` - (Required) The text of the SSH private key.

» Attributes Reference

- `id` The ID of the SSH key.

» tfe_team

Manages teams.

» Example Usage

Basic usage:

```
resource "tfe_team" "team" {
  name = "my-team-name"
  organization = "my-org-name"
}
```

» Argument Reference

The following arguments are supported:

- `name` - (Required) Name of the team.
- `organization` - (Required) Name of the organization.

» Attributes Reference

- `id` The ID of the team.

» tfe_team_access

Associate a team to permissions on a workspace.

» Example Usage

Basic usage:

```
resource "tfe_team" "team" {
  name = "my-team-name"
  organization = "my-org-name"
}

resource "tfe_workspace" "workspace" {
  name = "my-workspace-name"
  organization = "my-org-name"
}

resource "tfe_team_access" "access" {
  access = "read"
  team_id = "${tfe_team.team.id}"
  workspace_id = "${tfe_workspace.workspace.id}"
}
```

» Argument Reference

The following arguments are supported:

- `access` - (Required) Type of access to grant. Valid values are `admin`, `read` or `write`.
- `team_id` - (Required) ID of the team to add to the workspace.
- `workspace_id` - (Required) Workspace ID to which the team will be added.

» Attributes Reference

- `id` The team access ID.

» tfe_team_member

Add or remove a user from a team.

NOTE on managing team memberships: Terraform currently provides two resources for managing team memberships. The tfe_team_member resource can be used multiple times as it manages the team membership for a single user. The tfe_team_members resource, on the other hand, is used to manage all team memberships for a specific team and can only be used once. Both resources cannot be used for the same team simultaneously.

» Example Usage

Basic usage:

```
resource "tfe_team" "team" {
  name = "my-team-name"
  organization = "my-org-name"
}

resource "tfe_team_member" "member" {
  team_id = "${tfe_team.team.id}"
  username = "sander"
}
```

» Argument Reference

The following arguments are supported:

- `team_id` - (Required) ID of the team.
- `username` - (Required) Name of the user to add.

» tfe_team_members

Manages users in a team.

NOTE on managing team memberships: Terraform currently provides two resources for managing team memberships. The tfe_team_member resource can be used multiple times as it manages the team membership for a single user. The tfe_team_members resource, on the other hand, is used to manage all team memberships for a specific team and can only be used once. Both resources cannot be used for the same team simultaneously.

» Example Usage

Basic usage:

```
resource "tfe_team" "team" {
  name = "my-team-name"
  organization = "my-org-name"
}

resource "tfe_team_members" "members" {
  team_id = "${tfe_team.team.id}"
  usernames = ["admin", "sander"]
}
```

» Argument Reference

The following arguments are supported:

- `team_id` - (Required) ID of the team.
- `usernames` - (Required) Names of the users to add.

» Attributes Reference

- `id` - The ID of the team.

» tfe_team_token

Generates a new team token and overrides existing token if one exists.

» Example Usage

Basic usage:

```
resource "tfe_team" "team" {
  name = "my-team-name"
  organization = "my-org-name"
}

resource "tfe_team_token" "token" {
  team_id = "${tfe_team.team.id}"
}
```

» Argument Reference

The following arguments are supported:

- `team_id` - (Required) ID of the team.
- `force_regenerate` - (Optional) If set to `true`, a new token will be generated even if a token already exists. This will invalidate the existing token!

» Attributes Reference

- `id` - The ID of the token.
- `token` - The generated token.

» tfe_variable

Creates, updates and destroys variables.

» Example Usage

Basic usage:

```
resource "tfe_organization" "organization" {
  name = "my-org-name"
  email = "admin@company.com"
}

resource "tfe_workspace" "workspace" {
  name = "my-workspace-name"
  organization = "${tfe_organization.organization.id}"
}

resource "tfe_variable" "variable" {
  key = "my_key_name"
  value = "my_value_name"
  category = "terraform"
  workspace_id = "${tfe_workspace.workspace.id}"
}
```

» Argument Reference

The following arguments are supported:

- `key` - (Required) Name of the variable.
- `value` - (Required) Value of the variable.
- `category` - (Required) Whether this is a Terraform or environment variable. Valid values are `terraform` or `env`.
- `hcl` - (Optional) Whether to evaluate the value of the variable as a string of HCL code. Has no effect for environment variables. Defaults to `false`.
- `sensitive` - (Optional) Whether the value is sensitive. If true then the variable is written once and not visible thereafter. Defaults to `false`.
- `workspace_id` - (Required) ID of the workspace that owns the variable.

» Attributes Reference

- `id` - The ID of the variable.

» tfe_workspace

Provides a workspace resource.

» Example Usage

Basic usage:

```
resource "tfe_workspace" "my-workspace" {
  name = "my-workspace"
  organization = "my-organization"
}
```

» Argument Reference

The following arguments are supported:

- name - (Required) Name of the workspace.
- organization - (Required) Name of the organization.
- auto_apply - (Optional) Whether to automatically apply changes when a Terraform plan is successful. Defaults to false.
- terraform_version - (Optional) The version of Terraform to use for this workspace. Defaults to the latest available version.
- working_directory - (Optional) A relative path that Terraform will execute within. Defaults to the root of your repository.
- vcs_repo - (Optional) Settings for the workspace's VCS repository.

The vcs_repo block supports:

- identifier - (Required) A reference to your VCS repository in the format :org/:repo where :org and :repo refer to the organization and repository in your VCS provider.
- branch - (Optional) The repository branch that Terraform will execute from. Default to master.
- ingress_submodules - (Optional) Whether submodules should be fetched when cloning the VCS repository. Defaults to false.
- oauth_token_id - (Required) Token ID of the VCS Connection (OAuth Conection
  - Token) to use.