

ДИСЦИПЛИНА	Инструменты Девопс
ИНСТИТУТ	ИПТИП
КАФЕДРА	Индустриального программирования
ВИД УЧЕБНОГО МАТЕРИАЛА	Методические указания к практическим занятиям по дисциплине
ПРЕПОДАВАТЕЛЬ	Гиматдинов Дамир Маратович
СЕМЕСТР	5 семестр, 2025/2026 уч. год

Практическая работа №7.1

Задание на практическую работу

Инфраструктура в виде кода

Цель этого проекта - познакомить вас с основами инфраструктуры в виде кода (IaC) с использованием Terraform. Вы создадите репозиторий на GitHub и настроите его с помощью Terraform.

Требования

Вот требования к этой работе:

- Установить terraform:
 - С официального сайта: <https://developer.hashicorp.com/terraform/install>
 - Альтернативный вариант(использовать скрипт [build.sh](https://github.com/hashicorp/terraform) из папки scripts): <https://github.com/hashicorp/terraform>
- Вам необходимо написать скрипт Terraform, который создаст репозиторий на GitHub. Репозиторий должен быть общедоступным, и вы должны иметь возможность клонировать его с помощью SSH. Необходимо настроить с помощью Terraform параметры для GitHub репозитория, включая:
 - Название репозитория
 - Описание репозитория
 - Установку владельца репозитория
 - Установку лицензии репозитория
 - Добавление или удаление участников репозитория
 - Установку приватности репозитория
 - Установку ветки по умолчанию
 - Установку защиты ветки по умолчанию
 - Установку защиты коммитов

- o Установку защиты чтения и записи для файлов и папок в репозитории
- o Установку webhook'ов для автоматизации процессов в репозитории

Пример:

```

provider "github" {
  token = var.github_token
}

resource "github_repository" "example_repo" {
  name          = "example-repo"
  description   = "This is an example repository"
  visibility    = "public"

  owner = var.github_username

  default_branch = "main"

  allow_rebase_merge = true
  allow_squash_merge = true

  delete_branch_on_merge = true

  has_issues = true
  has_projects = true
  has_wiki = true

  allow_auto_merge = true

  template_repository = "github_repo/example-template"

  lifecycle {
    prevent_destroy = true
  }
}

resource "github_branch" "example_branch" {
  repository = github_repository.example_repo.name
  branch     = "main"
  protection {
    enabled = true
    required_pull_request_reviews {
      dismiss_stale_reviews = true
    }
    required_status_checks {
      contexts = ["ci/build"]
    }
  }
}

```

Подготовьте отчет по выполненной практической работе. В отчет добавьте скриншоты выполненных шагов.

Terraform — инструмент для развёртывания и управления инфраструктурой в виде кода (Infrastructure as Code, IaC). Разработан компанией HashiCorp. [nic.ruhostpro.by/ru.wikipedia.org*](https://nic.ruhostpro.by/ru.wikipedia.org/)

Особенности:

- Позволяет описывать ресурсы (виртуальные машины, сети, хранилища данных и другие) в виде конфигурационных файлов, написанных на языке HCL (HashiCorp Configuration Language). nic.ruhostpro.by
- Поддерживает множество облачных провайдеров, например Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP). [nic.ruru.wikipedia.org*](https://nic.ruru.wikipedia.org/)
- Подходит для управления локальной инфраструктурой, включая развёртывание виртуальных машин на базе VMware или настройку сетевых устройств. AdminVPS.ru

Принцип работы

Процесс работы Terraform включает несколько этапов: AdminVPS.ru

1. **Пользователь вводит желаемое состояние инфраструктуры** в конфигурационном файле.
2. **Команда terraform plan** проверяет, какие изменения нужны для приведения текущего состояния ресурсов в соответствие с указанным. На основании этой информации формируется план, показывающий, что именно будет создано, изменено или удалено.
3. **Команда terraform apply** применяет правки, создавая или модифицируя ресурсы.

Terraform также поддерживает управление текущим состоянием инфраструктуры через специальный файл состояния (state). AdminVPS.ru

Функции

- **Управление зависимостями** — инструмент позволяет определить зависимости между ресурсами, автоматически создавать и уничтожать ресурсы в правильном порядке. nic.ru
- **Повторяемость и воспроизводимость** — код Terraform можно сохранить в системе контроля версий (например, Git) и повторно использовать или восстановить в любое время. nic.ru
- **Расширяемость через модули** — Terraform позволяет объединять повторяющиеся части конфигурации в модули, которые можно вызывать в разных местах и проектах. practicum.yandex.ru
- **Интеграция с системами мониторинга и управления логами** — с помощью Terraform можно автоматически подключать создаваемые ресурсы к инструментам мониторинга, таким как Prometheus или Datadog. AdminVPS.ru

Установка

Terraform работает на большинстве операционных систем: Windows, macOS и Linux. **Чтобы установить инструмент, нужно:** practicum.yandex.ru

1. Перейти на официальный сайт: terraform.io.
2. Скачать соответствующую версию для своей операционной системы.
3. Распаковать архив и добавить путь к исполняемому файлу в переменную окружения PATH.
4. Проверить установку: terraform -v.

Сайт: terraform.io.

Репозиторий: github.com/hashicorp/terraform.

Последняя версия: 1.13.3 (на 17 сентября 2025 года). [ru.wikipedia.org*](https://ru.wikipedia.org)