COMPLETE V RISING APIS LIST

CORE FRAMEWORKS & LIBRARIES

1. BepInEx (Foundation)

- **Version:** 1.733.2
- **Type:** Plugin/Mod Framework
- **Purpose:** Core modding framework for Unity games
- **NuGet:** `BepInEx.Core`
- **Key Classes:**
  - `BasePlugin`
  - `BepInPlugin` (Attribute)
  - `BepInDependency` (Attribute)
  - `ManualLogSource`

---

2. VAMP (V Rising API Modding Platform)

- **Version:** 1.3.3+
- **NuGet:** `VRising.VAMP`
- **GitHub:** github.com/CrimsonMods/VAMP
- **Docs:** vrising.wiki

VAMP Services:

- **PlayerService -** Player data, inventory, equipment
- **CastleService -** Castle hearts, territories
- **SpawnService -** Entity spawning, NPC creation
- **TerritoryService -** Territory management
- **EventScheduler -** Global event management
- **FileReload -** Hot-reload configuration files
- **WipeData -** Automatic mod reset on wipes

- **ModTalk -** Inter-mod communication
- **ModSystem -** Thunderstore update notifications
- **ChatUtil -** Chat system extensions
- **ModProfiler -** Performance profiling

VAMP Data Classes:

- **VBloodData -** VBlood boss information
- **WorldRegionData -** World zones and regions
- **PrefabData -** Prefab collections

---

3. Bloodstone (Plugin Framework)

- **Version:** 0.2.3
- **NuGet:** `VRising.Bloodstone`
- **GitHub:** github.com/decaprime/Bloodstone
- **License:** MIT

Bloodstone Features:

- **VWorld -** World access helper
  - `VWorld.Server.EntityManager`
  - `VWorld.Client.EntityManager`
- **Plugin Hot-Reloading**
- **Keybinding Management**
- **IRunOnInitialized** interface
- **Initialization Hooks**

---

4. VampireCommandFramework (VCF)

- **Version:** 0.10.4
- **NuGet:** `VRising.VampireCommandFramework`

- **GitHub:** github.com/decaprime/VampireCommandFramework

VCF Core Classes:

- **CommandRegistry -** Command registration
- **ICommandContext -** Command execution context
- **CommandAttribute -** Command declaration
- **CommandGroupAttribute -** Group commands
- **ICommandMiddleware -** Middleware pipeline
- **CommandArgumentConverter<T> -** Type conversion

VCF Formatting:

- **Bold() -** Bold text
- **Italic() -** Italic text
- **Underline() -** Underline text
- **Color() -** Color text
- **Large() -** Large text
- **Small() -** Small text

VCF Built-in Commands:

- `.config list` - List all configurations
- `.config get` - Get configuration value
- `.config set` - Set configuration value

---

5. Bloody.Core (Helper Library)

- **Version:** 1.2.4+
- **NuGet:** `Bloody.Core`
- **Purpose:** Common utilities for Bloody ecosystem

Bloody Ecosystem Mods:

- **BloodyWallet -** Virtual currency system

- **BloodyEncounters -** Random NPC encounters
- **BloodyBoss -** Dynamic VBlood bosses
- **BloodyPoint -** Waypoint/teleportation
- **BloodyShop -** Merchant/shop system

---

6. VRisingServerApiPlugin (HTTP API)

- **Purpose:** REST API endpoints for server
- **Default Port:** 9090
- **Attributes:**
  - `[HttpGet]`
  - `[HttpPost]`
  - `[UrlParam]`
  - `[Body]`

UNITY & PROJECTM APIS

Unity.Entities (ECS Core)

Entity Management:

- **Entity -** Entity identifier
- **EntityManager -** Entity operations
  - `CreateEntity()`
  - `DestroyEntity()`
  - `Instantiate()`
  - `HasComponent<T>()`
  - `GetComponentData<T>()`
  - `SetComponentData<T>()`
  - `AddComponent<T>()`
  - `RemoveComponent<T>()`

- `GetComponentTypes()`

Queries & Systems:

- **EntityQuery -** Entity filtering
- **SystemBase -** System base class
- **ComponentSystemGroup -** System grouping
- **World -** ECS world container

ProjectM (V Rising Core Namespace)

Core Systems:

- **ServerGameManager -** Central game management
- **DebugEventsSystem -** Spawn items/buffs
- **ServerChatUtils -** Chat messaging
- **PrefabCollectionSystem -** Prefab management
- **DropItemThrowSystem -** Item dropping
- **ClanSystem -** Clan management
- **CastleHeartSystem -** Castle management
- **TerritorySystem -** Territory control
- **CraftingSystem -** Crafting operations
- **InventorySystem -** Inventory management
- **EquipmentSystem -** Equipment handling
- **BuffSystem -** Buff application
- **DamageSystem -** Damage processing
- **DeathSystem -** Death handling
- **RespawnSystem -** Respawning
- **TeleportSystem -** Teleportation
- **InteractSystem -** Entity interaction
- **MovementSystem -** Character movement

- **AbilityCastSystem -** Ability casting
- **HealthRegenSystem -** Health regeneration
- **BloodSystem -** Blood quality management
- **VBloodSystem -** VBlood boss tracking

Core Components:

- **User -** Player entity component
  - `PlatformId`
  - `CharacterName`
  - `IsConnected`
  - `IsAdmin`
- **Character -** Character data
  - `Entity`
  - `Name`
- **Health -** Health values
  - `Value` - Current health
  - `MaxHealth`
  - `IsDead`
- **Translation -** World position
  - `Value` (float3)
- **Rotation -** Entity rotation
  - `Value` (quaternion)
- **Equipment -** Equipped items
  - `WeaponSlot`
  - `ArmorSlots`
  - `CloakSlot`
  - `JewelrySlots`
- **Inventory -** Inventory data
  - `Items`

    - `Size`
- **Buff** - Buff data
    - `PrefabGUID`
    - `Target`
    - `Duration`
    - `Stacks`
- **Team** - Team affiliation
    - `Value`
    - `FactionIndex`
- **UnitLevel** - Entity level
    - `Level`
- **BloodConsumeSource** - Blood quality
    - `BloodQuality`
    - `BloodType`
- **Aggroable** - Can be aggro'd
- **Follower** - Follower data
- **Mount** - Mount data
- **CastleHeart** - Castle heart data
    - `CastleEntity`
    - `State`
- **UserOwner** - Owner reference
    - `Owner` (Entity)
- **PrefabGUID** - Prefab identifier
    - `GuidHash` (int)
- **Immortal** - Cannot die
- **Invulnerable** - Cannot take damage
- **DynamicBuffer<T>** - Dynamic data array
- **InventoryBuffer** - Inventory items

- **AbilityGroupSlotBuffer -** Ability slots
- **BuffBuffer -** Applied buffs
- **ModifyUnitStatBuff_DOTS -** Stat modifiers
- **VBloodConsumed -** VBloods killed
- **ProgressionMapper -** Progression data
- **AchievementInProgressElement -** Achievements
- **DeathEvent -** Death event data
- **StatChangeEvent -** Stat change events
- **CastStartedEvent -** Cast started
- **InteractEvent -** Interaction events
- **TakeDamageEvent -** Damage taken
- **DealDamageEvent -** Damage dealt

Network Components:

- **NetworkId -** Network identifier
- **FromCharacter -** Character reference
- **SpawnEntity -** Spawn request
- **SpawnPrefab -** Prefab spawn

World & Territory:

- **MapZone -** Map zone data
- **CastleTerritory -** Castle territory
- **TileModel -** Tile data
- **WorldRegionType -** Region types

STUNLOCK.CORE NAMESPACE

Core Utilities:

- **PrefabGUID -** Prefab identifier struct

- **NetworkId -** Network ID struct
- **ModificationId -** Modification tracking
- **FromCharacter -** Character source

UNITY.COLLECTIONS

Collections:

- **NativeArray<T> -** Native array
- **NativeList<T> -** Native list
- **NativeHashMap<K,V> -** Native hashmap
- **NativeQueue<T> -** Native queue
- **Allocator -** Memory allocation
  - `Allocator.Temp`
  - `Allocator.TempJob`
  - `Allocator.Persistent`

UNITY.MATHEMATICS

Math Types:

- **float2 -** 2D vector
- **float3 -** 3D vector
- **float4 -** 4D vector
- **quaternion -** Rotation
- **math -** Math utilities
  - `math.distance()`
  - `math.normalize()`
  - `math.dot()`
  - `math.cross()`
  - `math.lerp()`

HARMONY (PATCHING)

HarmonyLib:

- **[HarmonyPatch] -** Patch declaration
- **[HarmonyPrefix] -** Pre-execution patch
- **[HarmonyPostfix] -** Post-execution patch
- **[HarmonyTranspiler] -** IL modification
- **Harmony -** Harmony instance
  - `PatchAll()`
  - `Patch()`
  - `Unpatch()`

IL2CPP TYPES

IL2CPPChainloader:

- **Instance -** Chainloader instance
  - `Plugins` **-** Loaded plugins dictionary

IL2CPP Utilities:

- **Il2CppSystem.Collections.Generic -** IL2CPP collections

PREFAB GUID COLLECTIONS

Item PrefabGUIDs:

- Weapons, Armor, Resources, Consumables
- Structure components
- Crafting materials

Unit PrefabGUIDs:

- VBlood bosses

- NPCs and creatures
- Minions and summons

Buff PrefabGUIDs:

- Status effects
- Debuffs
- Passive bonuses

Ability PrefabGUIDs:

- Spells
- Vampire powers
- Weapon skills

Structure PrefabGUIDs:

- Buildings
- Crafting stations
- Furniture

COMMON HELPER EXTENSIONS

Entity Extensions:

```csharp
bool Exists(this Entity entity)
bool Has<T>(this Entity entity)
T Read<T>(this Entity entity)
void Write<T>(this Entity entity, T component)
void Add<T>(this Entity entity)
void Remove<T>(this Entity entity)
```

User Extensions:

```csharp
```

```
Entity GetCharacter(this Entity userEntity)
bool IsOnline(this Entity userEntity)
bool IsAdmin(this Entity userEntity)
void SendSystemMessage(this Entity userEntity, string message)
```

Character Extensions:

```
csharp

Entity GetUser(this Entity characterEntity)
float3 GetPosition(this Entity characterEntity)
void Teleport(this Entity characterEntity, float3 position)
void ApplyBuff(this Entity characterEntity, PrefabGUID buff)
void RemoveBuff(this Entity characterEntity, PrefabGUID buff)
```

CONFIGURATION APIS

BepInEx.Configuration:

- **ConfigFile** - Configuration file
  - `Bind<T>()` - Bind configuration entry
  - `Save()` - Save configuration
  - `Reload()` - Reload configuration
- **ConfigEntry<T>** - Configuration entry
  - `Value` - Current value
  - `SettingChanged` - Value changed event

LOGGING APIS

BepInEx.Logging:

- **ManualLogSource** - Log source
  - `LogInfo()`
  - `LogWarning()`
  - `LogError()`

- `LogDebug()`
- `LogMessage()`

COMMON PATTERNS

Entity Query Pattern:

```csharp
var query = EntityManager.CreateEntityQuery(
    ComponentType.ReadOnly<ComponentA>(),
    ComponentType.ReadWrite<ComponentB>()
);
var entities = query.ToEntityArray(Allocator.Temp);
```

Component Access Pattern:

```csharp
var component = EntityManager.GetComponentData<ComponentType>(entity);
component.Value = newValue;
EntityManager.SetComponentData(entity, component);
```

System Hook Pattern:

```csharp
[HarmonyPatch(typeof(SystemType), nameof(SystemType.OnUpdate))]
public static class SystemHook {
    public static void Prefix(SystemType __instance) {
        // Your code here
    }
}
```

Buff Application Pattern:

```csharp
```

```
var buffEntity = EntityManager.CreateEntity();
EntityManager.AddComponentData(buffEntity, new FromCharacter {
    Character = character,
    User = user
});
EntityManager.AddComponentData(buffEntity, new ApplyBuff {
    Target = target,
    BuffPrefabGUID = buffPrefab
});
```

ADDITIONAL RESOURCES

Documentation Sites:

- **VAMP Docs:** vrising.wiki
- **V Rising Mod Wiki:** wiki.vrisingmods.com
- **Unity DOTS:** docs.unity3d.com/Packages/com.unity.entities
- **BepInEx:** docs.bepinex.dev

Community:

- **Discord:** discord.gg/xzd5U5cNyD
- **Thunderstore:** v-rising.thunderstore.io
- **GitHub:** github.com/CrimsonMods, github.com/decaprime

Tools:

- **dotPeek:** Decompiler (jetbrains.com/decompiler)
- **.NET SDK 6.0:** Required for development
- **Visual Studio 2022:** Recommended IDE

INSTALLATION COMMANDS

```bash
```

```
# VAMP
dotnet add package VRising.VAMP

# Bloodstone
dotnet add package VRising.Bloodstone

# VampireCommandFramework
dotnet add package VRising.VampireCommandFramework

# Bloody.Core
dotnet add package Bloody.Core
```

This list represents ALL major APIs available for V Rising modding as of December 2025.