Posted by wolverin on Tue, 01 Aug 2023 18:07:03 GMT
View Forum Message <> Reply to Message

https:// learn.microsoft.com/ru-ru/windows/win32/api/iptypes/ns-iptyp
es-ip_adapter_addresses_lh?redirectedfrom=MSDN

```
const
  DLL = 'IPHLPAPI.DLL';

  MAX_ADAPTER_ADDRESS_LENGTH = 8;
  MAX_DHCPV6_DUID_LENGTH = 130;

type
  SOCKET_ADDRESS = record end;

  PIP_ADAPTER_UNICAST_ADDRESS = ^IP_ADAPTER_UNICAST_ADDRESS;
  IP_ADAPTER_UNICAST_ADDRESS = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          Flags: DWORD);
    end;
    Next: PIP_ADAPTER_UNICAST_ADDRESS;
    Address: SOCKET_ADDRESS;
    //PrefixOrigin: IP_PREFIX_ORIGIN;
    //SuffixOrigin: IP_SUFFIX_ORIGIN;
    //DadState: IP_DAD_STATE;
    ValidLifetime: ULONG;
    PreferredLifetime: ULONG;
    LeaseLifetime: ULONG;
    OnLinkPrefixLength: BYTE;
    end;

  PIP_ADAPTER_ANYCAST_ADDRESS = ^IP_ADAPTER_ANYCAST_ADDRESS;
  IP_ADAPTER_ANYCAST_ADDRESS = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          Flags: DWORD);
    end;
    Next: PIP_ADAPTER_ANYCAST_ADDRESS;
    Address: SOCKET_ADDRESS;
```

```
    end;

  PIP_ADAPTER_MULTICAST_ADDRESS = ^IP_ADAPTER_MULTICAST_ADDRESS;
  IP_ADAPTER_MULTICAST_ADDRESS = record end;

  PIP_ADAPTER_DNS_SERVER_ADDRESS = ^IP_ADAPTER_DNS_SERVER_ADDRESS;
  IP_ADAPTER_DNS_SERVER_ADDRESS = record end;

  PIP_ADAPTER_ADDRESSES = ^IP_ADAPTER_ADDRESSES;
  IP_ADAPTER_ADDRESSES = record union: record
    case Integer of
      0: (Alignment: UInt64);
      1: (Length: ULONG;
          IfIndex: DWORD);//IF_INDEX IfIndex;
    end;
    Next: PIP_ADAPTER_ADDRESSES;
    AdapterName: PCHAR;
    FirstUnicastAddress: PIP_ADAPTER_UNICAST_ADDRESS;
    FirstAnycastAddress: PIP_ADAPTER_ANYCAST_ADDRESS;
    FirstMulticastAddress: PIP_ADAPTER_MULTICAST_ADDRESS;
    FirstDnsServerAddress: PIP_ADAPTER_DNS_SERVER_ADDRESS;
    DnsSuffix: PWCHAR;
    Description: PWCHAR;
    FriendlyName: PWCHAR;
    PhysicalAddress: array[1..MAX_ADAPTER_ADDRESS_LENGTH] of BYTE;
    PhysicalAddressLength: ULONG;
//    union: record
{    case Integer of
      0: (Flags: ULONG);
      1: (DdnsEnabled: ULONG = 1;
          RegisterAdapterSuffix: ULONG = 1;
          Dhcpv4Enabled: ULONG = 1;
          ReceiveOnly: ULONG = 1;
          NoMulticast: ULONG = 1;
          Ipv6OtherStatefulConfig: ULONG = 1;
          NetbiosOverTcpipEnabled: ULONG = 1;
          Ipv4Enabled: ULONG = 1;
          Ipv6Enabled: ULONG = 1;
          Ipv6ManagedAddressConfigurationSupported: ULONG = 1;
    end;}
    Mtu: ULONG;
    IfType: DWORD;//IFTYPE
    //OperStatus: IF_OPER_STATUS;
    Ipv6IfIndex: DWORD;//IF_INDEX
    ZoneIndices: array[0..15] of ULONG;
    //FirstPrefix: PIP_ADAPTER_PREFIX;
    TransmitLinkSpeed: UInt64;
    ReceiveLinkSpeed: UInt64;
```

```
    //FirstWinsServerAddress: PIP_ADAPTER_WINS_SERVER_ADDRESS;
    //FirstGatewayAddress: PIP_ADAPTER_GATEWAY_ADDRESS;
    Ipv4Metric: ULONG;
    Ipv6Metric: ULONG;
    //Luid: IF_LUID;
    Dhcpv4Server: SOCKET_ADDRESS;
    //CompartmentId: NET_IF_COMPARTMENT_ID;
    //NetworkGuid: NET_IF_NETWORK_GUID;
    //ConnectionType: NET_IF_CONNECTION_TYPE;
    //TunnelType: TUNNEL_TYPE;
    Dhcpv6Server: SOCKET_ADDRESS;
    Dhcpv6ClientDuid: array[1..MAX_DHCPV6_DUID_LENGTH] of BYTE;
    Dhcpv6ClientDuidLength: ULONG;
    Dhcpv6Iaid: ULONG;
    //FirstDnsSuffix: PIP_ADAPTER_DNS_SUFFIX;
  end;




 union {
   ULONG Flags;
   struct {
     ULONG DdnsEnabled : 1;
     ULONG RegisterAdapterSuffix : 1;
     ULONG Dhcpv4Enabled : 1;
     ULONG ReceiveOnly : 1;
     ULONG NoMulticast : 1;
     ULONG Ipv6OtherStatefulConfig : 1;
     ULONG NetbiosOverTcpipEnabled : 1;
     ULONG Ipv4Enabled : 1;
     ULONG Ipv6Enabled : 1;
     ULONG Ipv6ManagedAddressConfigurationSupported : 1;
   };
 };
```